НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки.
Кафедра автоматики та управління в технічних системах.

«До захисту допущено»

Завідувач кафедри АУТС

_____ Ролік О.І.
(підпис) (ініціали, прізвище)

«__»_____2018р.

Магістерська дисертація

за напрямом підготовки Інформаційні системи та технології

на тему: Система управління міграцією віртуальних машин в корпоративній IT-інфраструктурі

Виконав: студент 6 курсу, групи ІА-з72мп
(шифр групи)

Величенко Андрій Вячеславович _____
(прізвище, ім'я, по батькові) (підпис)

Керівник зав. каф. АУТС д. т. н. професор Ролік О.І. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Рецензент _____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цьому дипломному проекті
немає запозичень з праць інших авторів
без відповідних посилань.

Студент _____
(підпис)

Київ 2018

Національний технічний університет України
“Київський політехнічний інститут
імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки
(повна назва)

Кафедра автоматики та управління в технічних системах
(повна назва)

Ступінь вищої освіти – другий (магістерський)
(код, назва)

Спеціальність 126 «Інформаційні системи та технології»
(код, назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Ролік О. І.
(підпис) (ініціали, прізвище)

“___”_____2018_р.

ЗАВДАННЯ

на магістерську дисертацію студенту

Величенку Андрію Вячеславовичу

(прізвище, ім’я, по батькові)

1. **Тема дисертації** Система управління міграцією віртуальних машин в корпоративній ІТ-інфраструктурі

Науковий керівник дисертації Ролік Олександр Іванович, проф. кафедри АУТС
затверджені наказом по університету від “29” жовтня 2018 р. №_____

2. Строк подання студентом дисертації: “4” грудня 2018р.
3. Об`єкт дослідження: віртуальні машини у складі гіпервізора в корпоративній ІТ-інфраструктурі.
4. Предмет досліджень є алгоритми, способи управління та реалізації системи управління міграцією віртуальних машин в корпоративній ІТ-інфраструктурі.
5. Перелік завдань, які потрібно розробити**:** провести огляд та аналіз існуючих рішень, яке програмне забезпечення та технології при цьому використовуються,

розробити алгоритми міграції віртуальних машин, розробити структурну схему системи управління.

6. Орієнтовний перелік ілюстративного (графічного) матеріалу: структурна схема, діаграма Use case, блок схеми алгоритмів міграції.
7. Орієнтовний перелік публікацій:
8. Консультанти розділів проекту:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

9. Дата видачі завдання “29” жовтня 2018_р.

Календарний план

№ з/п	Назва етапів виконання дипломного проекту	Термін виконання етапів проекту	Примітка
1	Огляд та аналіз існуючих рішень	27.10.2018	
2	Формування вимог та постановка задачі	06.11.2018	
4	Розробка структурної схеми	15.11.2018	
5	Розробка алгоритмів міграції	20.11.2018	
6	Розробка програмного забезпечення системи управління	23.11.2018	
7	Оформлення текстової та графічної документації	30.11.2018	
8	Представлення до захисту	4.12.2018	

Студент _____ Величенко А.В.
(підпис) (ініціали, прізвище)

Керівник проекту _____ Ролік О.І
(підпис) (ініціали, прізвище)

УДК 621.397.63

РЕФЕРАТ

Магістерська дисертація: 104 сторінок, 12 рисунків, 8 додатків, 20 джерел.

Актуальність теми підтверджуються великим попитом компаній на обчислювальні ресурси за низьку ціну.

Метою роботи є підвищення ефективності використання обчислювальних ресурсів пулу серверів в корпоративній IT-інфраструктурі, за рахунок алгоритмів міграції віртуальних машин.

Об’єктом дослідження є системи керування векторною тягою двигуна реактивного винищувача.

Предметом дослідження є алгоритми, способи управління та реалізації системи управління міграцією віртуальних машин в корпоративній IT-інфраструктурі.

Методологічну основу дослідження становлять фундаментальні положення комп’ютерної на програмної інженерії, наукові дослідження вітчизняних і зарубіжних компаній та вчених у сфері комп’ютеризованих систем. У магістерській дисертації проведено огляд існуючих рішень в корпоративних IT-інфраструктурах. Проведено аналіз і вивчення технології віртуалізації та систему управління гіпервізорами. Проаналізовано можливості створення системи управління міграцією. Розроблено структурну схему загальну та діаграму Use case, розроблено алгоритми міграції віртуальних машин.

Галузь застосування: Корпоративні IT-інфраструктури.

Ключові слова: віртуальна машина, система управління, система міграції, центр обробки даних, інфраструктура.

UDC 621.397.63

ABSTRACT

Master's thesis: 104 pages, 12 figures, 8 applications, 20 sources.

The relevance of the topic is confirmed by the great demand of companies for computing resources at a low price.

The aim of the work is to increase the efficiency of computing resources of the server pool in the corporate it infrastructure, through the software application of virtual machine migration algorithms.

The object of the study is the vector thrust control system of the jet fighter engine.

The subject of the research is algorithms, methods of management and implementation of the migration management system of virtual machines in the corporate it infrastructure.

The methodological basis of the research is the fundamental provisions of computer-aided software engineering, scientific research of domestic and foreign companies and scientists in the field of computerized systems. The master's thesis provides an overview of existing solutions in corporate it infrastructures. The analysis and study of virtualization technology and hypervisor management system have done. The possibilities of creating a migration management system are analyzed. The General block diagram and the diagram of Use case are developed, algorithms of migration of virtual machines are developed.

Scope: Corporate IT-infrastructure.

Keywords: virtual machine, control system, system migration, data center, infrastructure.

ЗМІСТ

Список скорочень

Administrator – Користувач з найбільшими привілеями

CPU – Центральний процесор (англ. – Central processing unit)

I/O – Ввід/Вивід (англ. – Input/Output)

NAS – Мережева система збереження даних (англ. – Network Attached Storage)

NAT– Перетворення мережних адрес (англ. – Network Address Translation)

NUMA – Архітектура пам'яті для багатопроцесорних систем (англ. – Non-Uniform Memory Access)

RAM – Оперативна пам’ять (англ. – Random Access Memory)

Root – Користувач з найбільшими привілеями у ОС сімейства Linux

SELinux – Linux з поліпшеною безпекою (англ. – Security-Enhanced Linux)

VPS – Віртуальний виділений сервер (англ. – Virtual Private Server)

ВМ – Віртуальна машина

ДБЖ – Джерело безперебійного живлення

ІТ – Інформаційні технологія

ОС – Операційна система

ПЗ – Програмне забезпечення

СЗД – Система зберігання даних

СУ – Система управління

ЦОД – Центр обробки даних

ВСТУП

У сучасному світі все більше набирає популярність хмарні технології. Дослідження Cisco/Intel [1] у якому приймало учать 4226 IT-директорів на підприємствах 18 галузей в дев'яти крупніших економік світу проведене у 2013 році наводить такі данні:

- у 2020 році очікуються 50 млрд пристроїв підключених до інтернету;
- витрати на хмарні технології у 2016 році складають 27% від всіх витрат на IT технології серед учасників опитування;
- приватні хмарні технології як модель використовуються в 45% випадках;
- на етапі експлуатації переваги хмарних технологій укладені в його можливостях по раціоналізації ряду IT-процесів, включаючи підтримку користувачів, а також поточні завдання контролю та обслуговування систем. У загальній складності 85 % співробітників, відповідальних за прийняття рішень у сфері IT, вважають, що хмарні технології позитивно вплинуть на експлуатацію і підтримку IT-систем, знизивши витрати і поліпшивши внутрішню доставку послуг;
- значна частка загального числа респондентів (82 %) також прогнозує, що хмара позитивно вплине на управління — заключний етап життєвого циклу IT-ресурсів. Але вони бачать зміни в управлінні як функції. 81 % респондентів вважають, що автоматизації відведено головне місце в процесах управління IT-ресурсами;
- в цілому, 70 % респондентів вважають, що єдиний постачальник «вкрай важливим» або просто «важливий». Проблема найбільш актуальна для Латинської Америки та Азіатсько-Тихоокеанського регіону (89 % і 85 % відповідно) і в меншій мірі турбує респондентів з Європи та Північної Америки з однаковим результатом — лише 55 %. Поліпшення функціональної сумісності між хмарними рішеннями також вважається набагато більш важливим в Азіатсько-Тихоокеанському регіоні, ніж в інших регіонах.

З дослідження [1] можна зробити висновок, що роль хмарних технологій приносить велику користь для компаній, які цими технологіями користуються. Але також слід зауважити, що є конкретні проблеми з переходом на хмарні технології наприклад робота сервісів та програмного забезпечення від різних виробників для різних платформ. Для вирішення цих проблем застосовуються технологія міграції.

Метою даної роботи є підвищення ефективності використання обчислювальних ресурсів пулу серверів в корпоративній IT-інфраструктурі, за рахунок програмного застосування алгоритмів міграції віртуальних машин. Розроблена система управління міграцією віртуальних машин дозволить більш раціонально застосовувати апаратні ресурси.

Об'єктом дослідження виступають технології віртуалізації комп'ютерних систем і відповідні програмні продукти, мережеві технології.

Призначення системи – управління процесом міграції віртуальних машин.
За допомогою цієї системи адміністратори центрів обробки даних зможуть більш якісно обслуговувати навантаження на сервери.

Область застосування – від центрів обробки даних до малих підприємств та установ, які бажають більш ефективно розпоряджатись своїми обчислювальними технічними ресурсами.

Ефективність системи управління полягає у використанні комбінацій різних алгоритмів міграції. Це дозволяє вирішити проблему простоїв та неефективного використання обчислювальних ресурсів. Також система управління міграцією дозволяє досягти низької щільності обчислювальних процесів на одиницю фізичного сервера, знімаючи необхідність в окремому сервері, виділеному для виконання тієї чи іншої користувальницької завдання.

1 АНАЛІЗ ІТ-ТЕХНОЛОГІЙ У КОРПОРАТИВНИХ ІТ-ІНФРАСТРУКТУРАХ

1.1 Аналіз корпоративної ІТ-інфраструктури

Корпоративна ІТ-інфраструктура – організаційно-технічне об'єднання програмних, обчислювальних і телекомунікаційних засобів, зв'язків між ними і експлуатаційним персоналом, що забезпечує обслуговування інформаційних, обчислювальних і телекомунікаційних ресурсів, можливостей та послуг працівникам необхідних для здійснення професійної діяльності та вирішення відповідних завдань.

ІТ-інфраструктура є не просто фундаментом для існування будь-якої сучасної організації, вона стає стратегічним активом, рушійною силою бізнесу. Також існують ІТ-інфраструктури, які можуть обслуговувати не тільки одну компанію, а багато компаній та користувачів одночасно. Такі ІТ-інфраструктури мають велику кількість обчислювальних ресурсів. Такі ІТ-інфраструктури називаються центрами обробки даних, або скорочено ЦОД. Для того, щоб побудувати надійну, масштабовану і високопродуктивну ІТ-інфраструктуру, потрібно задіяти велику кількість висококваліфікованих фахівців, які будуть володіти чималим досвідом.

Корпоративна ІТ-інфраструктура відрізняться від звичайного набору серверів такими ознаками:

- комплексне функціонування всіх частин інформаційної системи;
- промислова та функціональна сумісність;
- максимальний комфорт у використанні.

Корпоративна ІТ-інфраструктура займає особливе місце серед інструментів, які забезпечують функціонування бізнесу в сучасних умовах. Вона включає в себе всі ІТ-активи підприємства, апаратні та програмні засоби.

Мережеве обладнання, сервери, робочі станції, спеціалізовані програмні рішення і служби є компонентами ІТ-інфраструктури і потребують як якісної налаштуванні і супроводі, так і в постійному управлінні, модернізації та адаптації всієї інформаційної інфраструктури до потреб бізнесу та зміни зовнішніх умов.

Таким чином, корпоративна IT-інфраструктура – це не просто набір IT-рішень, вона являє собою велику інтегровану систему, котра забезпечує діяльність підприємства в цілому. Як і будь-яку систему, її необхідно цілеспрямовано проектувати і правильно експлуатувати, так як вона служить важливим елементом стратегії розвитку бізнесу. Інформаційна інфраструктура підприємства виступає основною рушійною силою розвитку бізнесу і розглядається як необхідна умова конкурентоспроможності та досягнення кінцевих цілей діяльності підприємства. Також у центрах обробки даних можуть проводитися наукові обчислення та моделювання які вимагають великої кількості обчислювальних ресурсів. Про Центри обробки даних з великим обсягом апаратного забезпечення надалі і піде мова.

1.2 Особливості використання ЦОД

Центр обробки даних з технічної точки зору – це сукупність обчислювальних ресурсів та ресурсів збереження даних, які об’єднані між собою мережевою інфраструктурою. З точки зору замовника, ЦОД дозволяє використовувати ресурси компанії власника ЦОД за привабливою ціною. У ролі замовника може виступати як сам користувач і напряму замовляти ресурси для власних потреб, так і компанії розробники програмного забезпечення, які бажають використовувати ресурси ЦОД для роботи їх власного програмного забезпечення. Розглянемо більш детально причини переходу від приватних обчислювальних інфраструктур до центрів обробки даних.

При використанні приватних серверів виникає дві очевидні проблеми стосовно гнучкості виконання ресурсів.

По-перше головна проблема це збільшення обчислювальних можливостей приватної інфраструктури. Часто бувають випадки коли потрібно збільшити можливості обчислювальних ресурсів, або ресурсів з збереження даних. Більш гостро ця проблема встає коли ці задачі потрібно вирішити терміново. В даному випадку компанії вимушені купляти сервери, обладнання з збереження даних,

мережеве та інше обладнання. Ці дії потребують часу як для планування зміни власної інфраструктури, так і для купівлі, доставки та встановлення цього обладнання. Центри обробки даних вирішують цю задачу маючи у своєму розпорядженні велику кількість вже встановленого та готового до роботи обладнання. Що у свою чергу допомагає компаніям швидко впроваджувати нові рішення та приймати нові виклики не хвилюючись про недостатню потужність власної комп'ютерної інфраструктури.

По-друге простоювання ресурсів які не використовуються у даний час. Це вимагає від підприємств витрачати більше коштів на підтримку власної IT інфраструктури. У той час як частина коштів яка витрачається на підтримку обладнання яке не використовується на повну потужність, могла бути використана у цілях розвитку компанії. Більш детально про недоліки у наступних пунктах.

Коли ЦОД більш ретельно використовують ресурси, приватні компанії у свою чергу вимушені також сплачувати більші рахунки на єлектроєнергію. Також компанія що володіє ЦОД зазвичай має спеціалізовані контракти з постачальниками єлектроєнергії що зумовлює меншу ціну на кВт·год. З використанням меншої кількості єлектроєнергії випливає наступний пункт.

Розвинені країни у всьому світі намагаються зберегти екологію світу. У ЦОД за рахунок більш ефективного використання ресурсів економлять єлектроєнергію і зберігають відповідний баланс між навколишнім середовищем, та індустріалізацією суспільства.

Компанії у конкурентній боротьбі за кожного клієнта постійно намагаються збільшити якість наданя власних послуг. Тому, окрім оновлення власної інфраструктури, компанії постійно оновлюють канали зв'язку до всесвітньої мережі інтернет. Маючи у своєму розпорядженні великі ресурси, власники ЦОД витрачають менше коштів на одиницю ресурсу. Також це дозволяє використовувати надійніші та більш якісні канали зв'язку (наприклад оптоволокно) та обладнання для доступу до мережі інтернет.

Компанії власники центрів обробки даних маючи велику кількість серверів, також мають більш вигідні контракти на ліцензування ПЗ на один сервер. Також

власники ЦОД як великі замовники ПЗ постійно співпрацюють з компаніями розробниками ПЗ. Що дозволяє мати більш якісне та спеціалізоване ПЗ. Це обумовлює наступний пункт.

Підприємства що мають власні ресурси витрачають кошти, окрім як на самі ресурси, ще і на обслуговування ресурсів. Системні адміністратори ЦОД маючи у своєму розпорядженні спеціалізоване ПЗ мають можливість обслуговувати більшу кількість серверів, ніж у компаніях що володіють приватними серверами.

Компанії власники ЦОД мають гнучку систему оплати за ресурси. Клієнти платять лише за ті ресурси які використовують. Це дозволяє клієнтам економити кошти, так як їм більше не потрібно обслуговувати ресурси якими вони не користуються. Також вони економлять свій власний час, отримуючи необхідну кількість ресурсів у тій кількості якій їм вона потрібна у будь-який час.

1.3 Огляд архітектури ЦОД

ЦОД складається з [17]:

– інформаційної інфраструктури, що включає в себе серверне обладнання та забезпечує обробку та зберігання інформації;

– телекомунікаційної інфраструктури, що забезпечує взаємозв'язок елементів ЦОД, а також передачу даних між ЦОД і користувачами;

– інженерної інфраструктури, що забезпечує нормальне функціонування основних систем.

– Інженерна інфраструктура включає в себе: кондиціонер для підтримки температури і рівня вологості в заданих параметрах;

– безперебійне електропостачання для автономної роботи у випадках відключення центральних джерел електроенергії;

– охоронно-пожежну сигналізацію і система газового пожежогасіння;

системи віддаленого IP-контролю, керування живленням та контролю доступу.

Деякі ЦОД пропонують клієнтам додаткові послуги по використанню обладнання з автоматичного відходу від різних видів атак. Команди кваліфікованих

фахівців цілодобово проводять моніторинг усіх серверів. Необхідно зазначити, що послуги ЦОД сильно відрізняються в ціні і кількості послуг. Для забезпечення збереження даних використовуються системи резервного копіювання. Для запобігання крадіжки даних, у використовуються різні системи обмеження фізичного доступу, системи відеоспостереження. Обладнання кріпиться в спеціалізованих стійках і шафах. Як правило, в ЦОД приймають для розміщення лише обладнання в стійках, тобто у корпусах стандартних розмірів. Комп'ютери в корпусах настільного виконання незручні і розміщуються в них рідко.

1.4 Аналіз технологій віртуалізації у ЦОД

За для збільшенню ефективності використання обчислювальних ресурсів використовують технології віртуалізації. Технології віртуалізації займають одне з найбільших місць у центрах обробки даних.

Технологія віртуалізації використовується на всіх рівнях ЦОД:

– фізичний рівень. Віртуалізація систем збереження даних (NAS, SAN), мережі передачі даних (VPN, VLAN);

– рівень сервісу. Віртуалізація операційних систем (XEN, Hyper-V);

– рівень користувачів. Додатки написані за допомогою технологій Java, Microsoft .NET.

В даний час існують наступні типи віртуалізації:

– гостьова операційна система;

– паралельна віртуальна машина;

– на основі гіпервізора;

– віртуалізація на рівні ядра.

Можна зробити висновок що технології віртуалізації будуть і надалі поширюватись у ЦОД незалежно від того чи обслуговує він одне підприємство, чи це ЦОД у якому орендуються ресурси користувачами. У цій дипломній роботі буде йти мова про міграцію віртуалізованих технологій рівня сервісу.

1.4.1 Аналіз віртуальної машини

Віртуальна машина це емуляція програмного забезпечення для його нормального функціонування у апаратному середовищі. У віртуальній машині підлягає емуляції все, починаючи від ядра операційної системи і драйверів для роботи з апаратним забезпеченням, закінчуючи додатками користувача.

Основні задачі які були поставлені розробниками віртуальних машин наступні:

- забезпечення нормального функціонування додатків в межах ВМ;
- підтримка різних фізичних архітектур;
- захищеність додатків та супутньої інформації;
- оптимізація використання ресурсів фізичних серверів.

Віртуальна машина представляє собою операційну систему, яка має власне ядро, драйвери для роботи з фізичним обладнанням, додатки користувача, власні налаштування як і у звичайних операційних систем. ВМ не може самостійно виконувати обчислювальні процеси та взаємодіяти з фізичними пристроями.

Для виконання віртуальною машиною користувальницьких завдань та забезпечення її функціонування фактично використовуються обчислювальні ресурси реальної системи, в середовищі якої запущена ця віртуальна машина. Віртуальна машина є всього лише набором програм, які виконуються в системі у виділеній на період існування машини області оперативної пам'яті.

Також зазначимо декілька можливостей ВМ при їх повній ізоляції від зовнішнього світу.

По-перше, різні інструменти віртуалізації дозволяють віртуальним машинам одержувати доступ до деяких користувальницьким пристроїв наприклад, USB-накопичувачів, розділів жорстких дисків або навіть цілим дисків.

По-друге, віртуальні машини нерідко включаються в існуючі комп'ютерні мережі як повноцінних членів, або навіть особливі віртуальні мережі, що складаються виключно з віртуальних машин, чия робота фізично відбувається лише в оперативній пам'яті фізичного середовища.

По-третє, процесори останніх поколінь надають спеціальні машинні команди і структури даних для створення віртуальних машин і управління ними.

Проте слід розуміти, що без налаштованої явно комунікації із зовнішнім світом за допомогою мережі загального дискового простору або чого-небудь ще, віртуальна машина повністю замкнута у виділеній їй області пам'яті і не може отримати доступу до процесів або даними як фізичного середовища, так і інших віртуальних машин. Адресний простір її оперативної пам'яті і межі відведеного для роботи ВМ дискового простору суворо обмежені, незважаючи на те, що фізично ця машина ділить з собі подібними і фізичним обчислювачем якісь єдині ресурси.

1.4.2 Аналіз віртуалізації на основі гіпервізора

При використанні цього підходу в архітектурі системи з'являється додатковий елемент – так званий гіпервізор, що представляє собою легковажну і високо оптимізовану програмну прошарок між апаратним і програмним забезпеченням. У термінах концепції кілець захисту, гіпервізор виконується в нульовому кільці, див. Рисунок 1 .Хоча, з таким рівнем привілеїв виконується операційна система, у свою чергу, гостьові операційні системи і віртуальні машини виконуються на менш привілейованих рівнях.

Гіпервізор виконує кілька основних завдань.

По-перше, він здатний пізнавати, перехоплювати і відповідати на так звані привілейовані команди. Це особливий клас машинних команд, які можуть бути породжені тільки ядром операційної системи. Ці команди потім емулюються гіпервізором, як якщо б вони були виконані від імені відповідної операційної системи. Цей прийом називається зниженням рівня привілеїв кільця.

По-друге, він відповідає за впорядкування, перенаправлення та повернення результатів запитів від операційних систем до обладнання. Керуюча операційна система виконується в середовищі гіпервізора, так само, як і всі віртуальні машини, вона використовується для управління гіпервізором і віртуальними машинами.

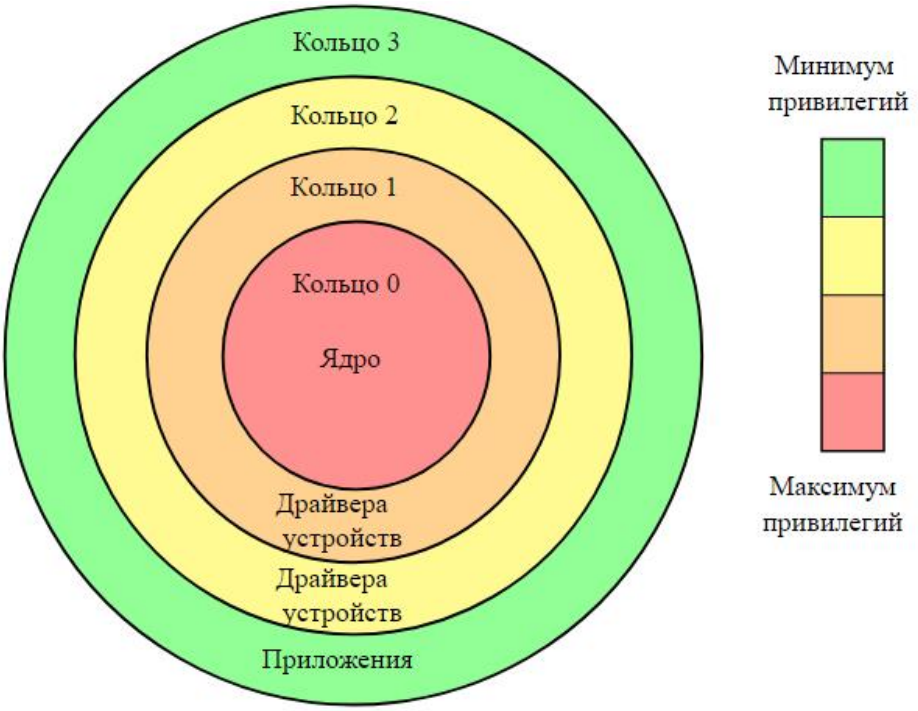


Рисунок 1.1 – Рівні привілеїв [3]

Таким чином, гіпервізор розширює ідею спільного використання ресурсів безліччю користувальницьких завдань і вирішує завдання планування процесорного часу подібно до того, як це робить будь-яка операційна система. Тільки для гіпервізора споживачами ресурсів є не окремі програми, а цілі операційні системи, що, звичайно ж, ускладнює завдання.

1.5 Хмарні технології

Хмарні технології – результат розвитку архітектур інформаційних систем. Під терміном "хмарні технології" будемо розуміти модель, яка забезпечує багатокористувацький доступ до обчислювальним ресурсам за допомогою розподіленої гетерогенної мережевої інфраструктури. Управління завантаженням обчислювальні ресурси здійснюється оперативно за запитом користувача при мінімальних експлуатаційних витратах. Принциповою особливістю від є акцент на переважаючому самообслуговуванні його користувачів. Як показує аналіз розвитку on-line сервісів, обсяг застосування неухильно зростає. Доступ до необхідних

користувачеві обчислювальні ресурси забезпечує хмарний провайдер. Якщо він є сторонньою організацією і ресурси надаються за допомогою інтернет - хмара буде називатися публічним.

При такому підході повне технічне обслуговування затребуваних ресурсів здійснюється сторонньою організація. При цьому кількість ресурсів в хмарі є умовно нескінченним – споживач отримує рівно стільки ресурсів, скільки запитує. Якщо хмарним провайдером є IT відділ організації, а ресурси надаються всередині іс, то хмара є приватним. Приватна хмара, це хмара організації, яка для користувачів виглядає як публічний. Воно володіє всіма перевагами приватного хмари, але функціонує на базі локального ЦОД. Отакий підхід набув найбільшого поширення в даний час. Він дозволяє зберегти користувачам повний контроль над своїми даними, які не завжди можливо розмістити в публічній хмарі за вимогами безпеки та конфіденційності.

При створенні приватної хмари IT інфраструктура організації централізується і віртуалізується. Тобто все обчислювальні ресурси збираються в один обчислювальний кластер, потім діляться на логічні розділи і видаються користувачам. Таким чином загальні апаратні ресурси організації можуть бути використані для різних завдань. Як показує статистика, ВМ демонструють продуктивність на рівні аналогічних апаратних платформ. Впровадження приватної хмари зменшує складність IT інфраструктури, підвищує її керованість, а також дозволяє оптимізувати використання наявних ресурсів. Економія в хмарній моделі досягається за рахунок ефективного використання поділюваного пулу серверів. Оптимізація можлива і за рахунок централізованого адміністрування-якість управління підвищується, число системних адміністраторів зменшується.

Існують також громадські хмари для використання спільнотами користувачів або організацій, а також гібридні хмари. Гібридною хмарою називається комбінація приватного і публічної Хмари-організація використовує свою приватну хмару, але при нестачі власних ресурсів додаткові потужності замовляються в публічних хмарах.

Програмне забезпечення як послуга реалізовують коли користувачеві надається програмне забезпечення. Наприклад, додатки провайдера, що виконуються на хмарі. В використовується різними клієнтами в гетерогенному середовищі. Споживач не управляє і не контролює саму хмарну інфраструктуру, на якій виконується додаток, будь то мережі, сервери, операційні системи, системи зберігання або навіть деякі специфічні для додатків можливості. У ряді випадків, споживачеві може бути надана можливість доступу до деяких користувальницьких конфігураційних налаштувань.

Платформа як послуга. Використовується розробниками та інтеграторами ПЗ. Споживачеві надаються засоби для розгортання на хмарній інфраструктурі створюваних або придбаних програм, що розробляються з використанням підтримуваних провайдером інструментів і мова програмування.

Інфраструктура як послуга застосовується коли споживачеві надаються кошти обробки даних, зберігання, мереж та інших базових обчислювальні ресурси, на яких споживач може розгортати і виконувати довільне ПЗ, включаючи операційні системи і додатки. Споживач не управляє і не контролює саму хмарну інфраструктуру, але може контролювати операційні системи, засоби зберігання, розкривні Додатки і, можливо, володіти обмеженим контролем над обраними мережевими компонентними. Наприклад, мережевий екран хоста, керованого споживачем.Кожна з перерахованих сервісних моделей може мати свої підвиди, в залежності від типу послуг.

Одну з актуальних завдань підприємств вирішує сервісна модель робочий стіл як сервіс – DaaS. DaaS є одним з видів приватного хмари сервісної моделі IaaS. Якщо в класичному IaaS зазвичай надаються ресурси для розміщення серверних додатків, то DaaS надає ресурси для використання графічного інтерфейсу користувача операційних систем, розміщених на сервері. DaaS базується на відповідній IT інфраструктурі-інфраструктурі віртуальних робочих місць або VDI (Virtual Desktop Infrastructure).

Традиційно кожне робоче місце являє собою персональний комп'ютер і набір периферійних пристрій. Такий підхід має ряд недоліків. Старіння парку

персональних комп'ютерів і необхідність його заміни кожні 4-5 років, високі витрати обслуговування, кожен персональний комп'ютер необхідно окремо налогоджувати, розміщення ресурсів при відсутності співробітника на робочому місці, відсутність доступу до даних поза робочим місцем.

На зміну традиційному принципу організації робочого простору приходить концепція DaaS. Віртуалізація робочого місця це поділ персонального комп'ютера на дві частини: серверну та клієнтський. Основна функціональна частина переноситься на сторону сервера. На клієнтській частині залишається тільки периферія. Тобто, в найбільш поширеному варіанті, перед кінцевим користувачем залишається монітор, мишка, клавіатура і невеликий пристрій для підключення перерахованого вище до мережі. При цьому користувач може підключитися до комп'ютера не тільки зі свого робочого місця, але і з будь-якого пристрою підключеного до мережі інтернет. Такий підхід має низку переваг перед класичним використанням персональних комп'ютерів. спрощення адміністрування, коли кожне робоче місце є копією одноразово налаштованого віртуального комп'ютера.

Підвищення мобільності співробітників, коли архітектура віртуальних робочих місць дозволяє налаштувати віддалене підключення до будь-якого віртуального комп'ютера в корпоративній мережі. Спрощення оновлення обладнання, коли при заміні серверного обладнання є можливість зберегти всі персональні віртуальні комп'ютери всіх співробітників в незмінному вигляді. Економія ресурсів, так як за кожним віртуальним комп'ютером не закріплені певні фізичні ресурси, що дозволяє використовувати менше обчислювальних ресурсів, ніж це необхідно для вихідної кількості персональних комп'ютерів.

Для створення DaaS хмари необхідно створити відповідну VDI інфраструктуру. Так як, обчислювальні ресурси в хмарі виділяються користувачами самостійно, передбачати необхідні обчислювальні ресурси дуже складно. Крім того, кожна ВМ може споживати різну кількість ресурсів, в залежності від її навантаження. Для визначення споживаних ВМ ресурсів застосовують алгоритми передбачення навантаження.

Також в даному випадку відіграє важливу роль організація системи зберігання даних і алгоритм розміщення віртуальних машин на обчислювальних серверах. Чим щільніше будуть розташовуватися ВМ, тим менше кількість фізичних серверів потрібна. Щільне розташування ВМ на серверах дозволить також вимикати НД при низькому завантаженні, що забезпечить режим енергоефективності. Основною проблемою є автоматичне виділення оптимальної кількості апаратних ресурсів в комплексі віртуалізації для забезпечення ефективної і безперебійної роботи хмари.

Першочерговим завданням при проектуванні Хмари віртуальних робочих місць є оцінка потреби в обчислювальних ресурсах. Для виконання заявленого завдання, на першому етапі, визначимо структуру проектованого комплексу.

У складі хмари можна виділити програмну і апаратну складові. Апаратними компонентами є: системи зберігання даних, обчислювальні сервера, мережева інфраструктура і клієнтські пристрої. Програмні компоненти є: ПЗ керування СГД, монітор віртуальних машин, ПЗ керування монітором віртуальних машин, клієнтське ПО віддаленого доступу.

Для запуску додатків користувачі з клієнтських пристроїв віддалено підключаються до віртуальних машин, запущеним на обчислювальних серверах. При цьому файли віртуальних машин, віртуальні жорсткі диски, зберігаються в системі зберігання даних. Обмін даними між компонентами системи забезпечується за допомогою мережевої інфраструктури, що складається з комутаційного обладнання комутатори, або маршрутизатори, і середовища передачі.

На клієнтських місцях має бути встановлено спеціальне ПЗ для віддаленого доступу до віртуальних машина. Роботу віртуальних машин забезпечує монітор ВМ, встановлений на ВС. Окремі компоненти монітора ВМ можуть також забезпечувати балансування навантаження між ВС. Дані ВМ зберігаються на системі зберігання даний. Звернення до цих даних від монітора ВМ обробляються контролерами СГД.

Для забезпечення відмовостійкості застосовують кілька НД, які замінюють один одного в разі відмови. При цьому, ВМ запущені на одному сервері, мігрують на інший без зупинки. Хоча насправді робота ВМ в цьому випадку призупиняється на кілька секунд. СГД так само повинна містити кілька контролерів, для

безперебійної обробки запитів від ВС. Дані в СГД зберігаються в RAID масивах, що дозволяє зберегти дані, навіть якщо деякі диски вийшли з ладу.

Особливістю приватних хмар орієнтованих на роботу з віртуальними робочими столами, є великий обсяг графічної інформації, що передається від ВС до клієнтського пристрою. Таким чином, при проектуванні приватного хмари з наданням робочого столу, необхідно приділити окрему увагу пропускної здатності мережевої інфраструктури.

Впровадження хмарних технологій в IT інфраструктуру підприємств дозволяє значно скоротити витрати на її супровід за рахунок оптимізації використання апаратних ресурсів та спрощення адміністрування. Хмара віртуальних робочих місць (DaaS) в даному випадку не є винятком-його впровадження призведе до віртуалізації робочих місць і централізації IT інфраструктури. При переході організації від класичних персональних комп'ютерів до віртуальних робочих місць необхідно побудувати інфраструктуру віртуальних робочих місць (VDI). При проектуванні VDI Хмари необхідно вирішити задачу оцінки необхідного числа вузлів кластера віртуалізації на якому будуть розташовуватися віртуальні машини користувачів. Іншим фактором, від якого залежить необхідна кількість вузлів є алгоритм розміщення віртуальних машин на ВС. Вибір оптимального алгоритму визначає найбільш щільне розміщення ВМ, що призведе до зменшення необхідного числа вузлів. Для того щоб врахувати перераховані вище умови в роботі запропонована модель оцінки необхідного кількості вузлів кластера віртуалізації, заснована на вирішенні задачі про оптимальну упаковці. Використання такого апарату стає життєво важливим при проектуванні хмарних платформ з великим обсягом споживання. Даний алгоритм визначення необхідної кількості вузлів універсальний і може застосовуватися при рішення задач проектування хмарних платформ самого різного призначення

1.6 Підсумки аналізу IT-технологій у корпоративних IT-інфраструктурах

Концентрація великої кількості ресурсів в одному місці, використовуючи при цьому стандарти які були наведені у підрозділах 1.5.1 та 1.5.2 дозволяє економити кошти на обслуговуючому персоналі, системах кондиціонування, електроенергії та системах контролю енергопостачання. Також стає можливим вирішення задач, які потребують велику кількість обчислювальних ресурсів. Наприклад, задачі з сфери охорони здоров’я, природничих наук, військової сфери. Маючи можливість наведену у підрозділі 1.3 орендувати ресурси, компанії, або звичайні користувачі можуть швидко отримати доступ до великого обсягу обчислювальної техніки за дуже короткий обсяг часу, що підвищує їх продуктивність і сприяє розвитку всіх напрямків суспільства. Але виникає проблема у керуванні великою кількістю ресурсів в IT-інфраструктурі та ефективного способу використання ресурсів. Ефективно використовувати ресурси ЦОД можна завдяки збільшенню щільності виконання багатьох додатків різних користувачів використовуючи технології віртуалізації наведені у підрозділі 1.6.

2 РОЗРОБКА СТРУКТУРИ СИСТЕМИ УПРАВЛІННЯ ВМ

2.1 Проектування об’єкту управління

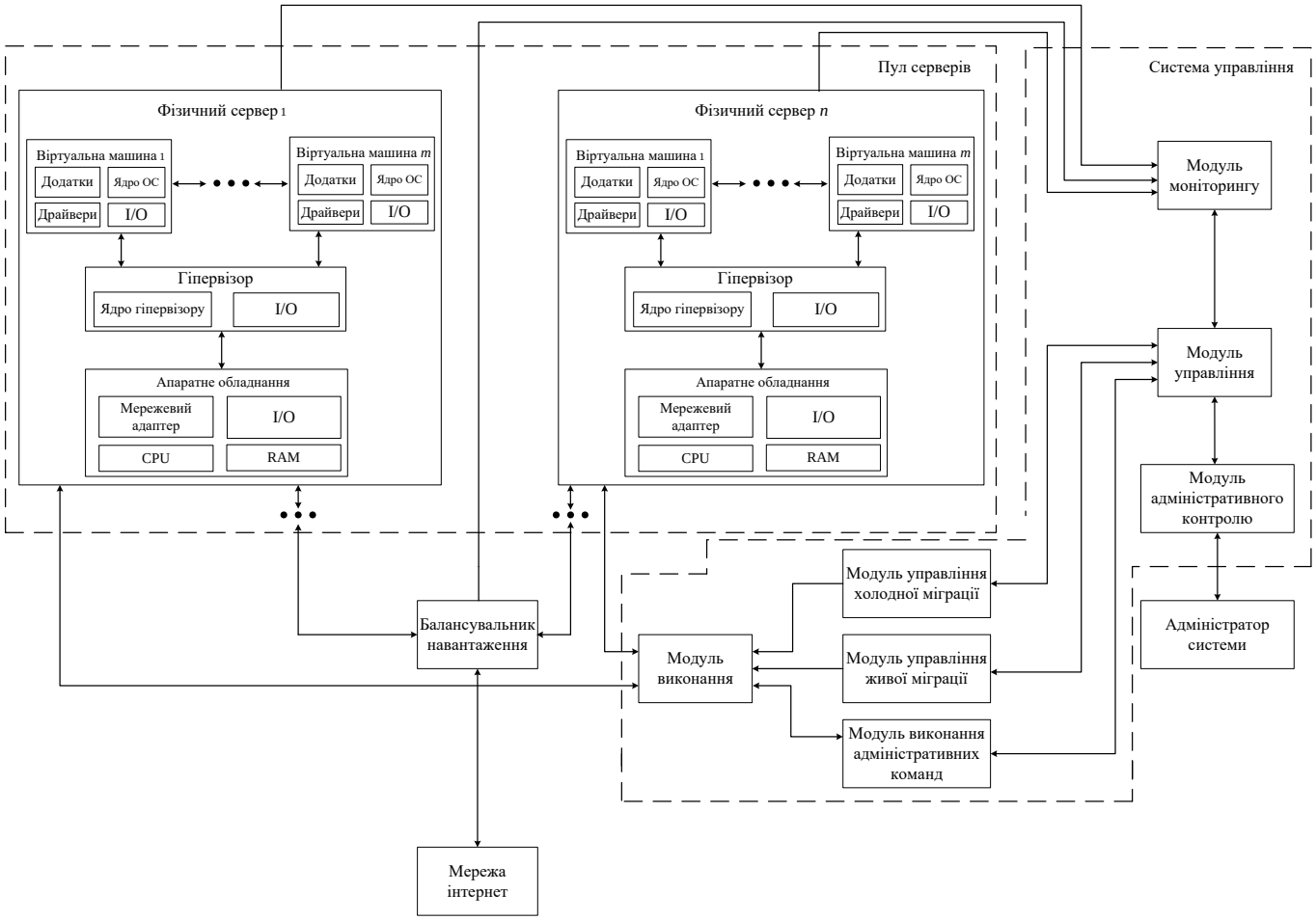
Для початку буде розглянуто, що з собою представляє одиничний сервер, з яких компонентів складеться і для яких типів задач може використовуватись. А також у наступних розділах буде розглянуто, як система управління ВМ вирішує проблему міграції в залежності від задач які виконує сервер.

Загальну структурну схему системи управління вм наведено у додатку А.

На структурній схемі зображені компоненти фізичного серверу у пулі серверів, систему управління віртуальними машинами, адміністратора системи який може давати команди системі та отримувати сповіщення про результат виконаної роботи а також мережу інтернет через яку користувачі розміщують свої додатки.

Пул серверів – це об’єднання множини серверів каналом зв’язку. Сервера об’єднуються в пул для зручності адміністрування, кластерних обчислювань та збільшення швидкодії між фізичними серверами які знаходяться в одному пулі. У пулі серверів, як прийнято, знаходяться сервери з однаковим фізичним устаткуванням. Фізичні сервери отримують додатки користувачів через балансувальник навантаження, а користувачі в свою чергу надсилають свої додатки через мережу інтернет до якої підключений балансувальник навантаження. Для користувачів логіка роботи корпоративної інфраструктури не відома, вони працюють з своїми додатками так наче додатки розташовані на окремому сервері до якого користувачі під’єднуються напряму.

Система управління міграцією віртуальних машин як раз була розроблена для управлінням процесом міграції ВМ між одиницями серверів які об’єднані в пул серверів без контакту з користувачами додатків. Це дозволяє більш якісно та оптимально використовувати фізичні ресурси. І економити час користувачів додатків.



Додаток А – структура пулу серверів системи управління ВМ в ЦОД

2.2 Проектування фізичного устаткування одиничного серверу

Фізичний сервер у центрах обробки даних складеться з наступних компонентів:

– CPU. Один фізичний сервер може мати декілька CPU, але як найменше один завжди присутній. Кількість CPU в одному фізичному сервері варіюється в залежності від архітектури самого CPU. Система управління міграцією ВМ розрахована на роботу CPU архітектури сімейства x86 та сумісних з нею архітектур. На фізичних серверах архітектури сімейства x86 кількість CPU може доходити до восьми. Але зазвичай використовують фізичні системи з двома, або чотирьома CPU. Також для біль ефективного використання фізичних ресурсів віртуальними

машинами застосовуються інструкції апаратної віртуалізації Intel VT, або AMD-V в залежності від виробника CPU.

- RAM. Кількість оперативної пам’яті залежить від задач які повинні виконувати сервер. Максимальна кількість оперативної пам’яті залежить від виробника материнських плат, та від контролера RAM який встановлений в CPU. Необов’язковою вимогою є використання технології ECC. Ця технологія дозволяє знаходити та виправляти помилки у RAM ще на апаратному рівні забезпечуючи таким чином більш стабільну роботу всій апаратній системі.
- Мережевий адаптер. Кількість мережевих адаптерів та протоколи які використовуються для передачі інформації в каналах зв’язку відрізняються в залежності від задач, які має виконувати фізичний сервер, та від вже існуючої архітектури в центрах обробки даних. Це можуть бути протоколи сімейства Ethernet які описуються стандартами IEEE групи 802.3, або протокол Fibre Channel який описується стандартом технічного комітету T11 у складі INCITS, акредитованого ANSI. Система управління міграцією ВМ розрахована на передачу даних використовуючи протоколи сімейства Ethernet. До мережевого адаптеру підключається також система зберігання даних.

Окрема також слід зазначити інтерфейс I/O. За допомогою цього інтерфейсу відбувається передача інформації між фізичними та логічними компонентами системи.

2.3 Використання гіпервізорів

Компоненти гіпервізорів про які буде йти мова можна логічно розділити на дві групи:

- ядро гіпервізору, яке виконує службові функції;
- I/O який відповідає за передачу інформації .

Обговоримо кожну систему віртуалізації на основі гіпервізора.

2.3.1 Використання гіпервізора Microsoft Hyper-V

Компанія Microsoft розробила комплекс програмного забезпечення до якого входить сам гіпервізор та система управління до нього. Він може працювати як окреме рішення, так і як частина операційної системи Microsoft Windows Server. Компоненти в архітектурі Hyper-V взаємодіють між собою користуючись таким поняттям як розділ. Розділ – це логічне поняття яке дозволяє розмежувати об’єкти системи. Розділи бувають двох типів кореневий і дочірні. Тільки однин кореневий розділ може бути на одному фізичному сервері. У той час як кількість дочірніх розділів може доходити до 1024

До кореневого розділу Hyper-V входить:

- низькорівнева оболонка Microsoft Windows;
- служба управління ВМ Hyper-V;
- компонент віртуалізації WMI;
- компонент віртуалізації шини VMbus;
- служба віртуалізації VSP;
- драйвер віртуальної інфраструктури VID.

Стек віртуалізації запускається на кореневому розділі і має прямий доступ до апаратних пристроїв. Потім кореневий розділ породжує дочірні розділи, на яких і розташовуються гостьові ОС. Дочірній розділ також може породити власні дочірні розділи. Кореневий розділ створює дочірні за допомогою API-гіпервізора, представленого в Hyper-V.

Під кожну нову гостьову операційну систему створюється новий дочірній розділ. Операційні системи знаходячись кожна в своєму розділі не мають прямого доступу як одна до одній, так і до апаратних ресурсів фізичного серверу. Виконання всіх програмних команд гостьових ОС йде через кореневий розділ.

Віртуалізовані розділи не мають ні доступу до фізичного процесора, ні можливості управляти його реальними перериваннями. Замість цього у них є віртуальне уявлення процесора і гостьова віртуальний адреса, яка залежить від конфігурації гіпервізора. Віртуальна адреса зовсім необов'язково при цьому займає

весь віртуальний адресний простір. Гіпервізор може визначати кількість логічних ядер одного процесора, або декількох процесорів для кожного розділу. Гіпервізор управляє перериваннями процесора і перенаправляє їх у відповідний розділ, використовуючи логічний контролер штучних переривань. Hyper-V може апаратно прискорювати трансляцію адрес між різними гостьовими віртуальними адресними просторами за допомогою пристрою керування вводу-виводу пам'яті, яке працює незалежно від апаратного управління пам'яттю, яка використовується процесором.

Дочірні розділи не мають безпосереднього доступу до апаратних ресурсів, але зате отримують віртуальне уявлення ресурсів, і працюють з ними як з віртуальними пристроями. VMBus-це логічний канал, що здійснює взаємодію між розділами. Будь-яка спроба звернення до віртуальних пристроїв перенаправляється через VMBus до пристроїв батьківського розділу, які і оброблять даний запит. Відповідь повертається також через VMBus. Якщо пристрої кореневого розділу також є віртуальними пристроями, то запит буде передаватися далі, поки не досягне такого кореневого розділу, де він отримає доступ до фізичних пристроїв. Кореневі розділи запускають провайдер сервісу віртуалізації, який з'єднується з VMBus і обробляє запити доступу до пристроїв від дочірніх розділів. Віртуальні пристрої дочірнього розділу працюють з клієнтом сервісу віртуалізації, який перенаправляє запит через VMBus до кореневого розділу. Цей процес прозорий для гостьової ОС.

Віртуальні пристрої також підтримують технологію Windows Server Virtualization, яка має назву прогресивне введення-виведення , для накопичувачів, мережевих і графічних підсистем в тому числі. прогресивне введення-виведення — спеціалізована віртуалізована реалізація високорівневих протоколів, наприклад, SCSI, для можливості працювати з VMBus безпосередньо, що дозволяє паралельно обробляти будь-які рівні емуляції пристрою. Це робить взаємодію більш ефективною, але натомість вимагає від гостьової ОС підтримки прогресивне введення-виведення.

Взаємодію компонентів Microsoft Hyper-V можна побачити на рисунку 2.2 [4]:

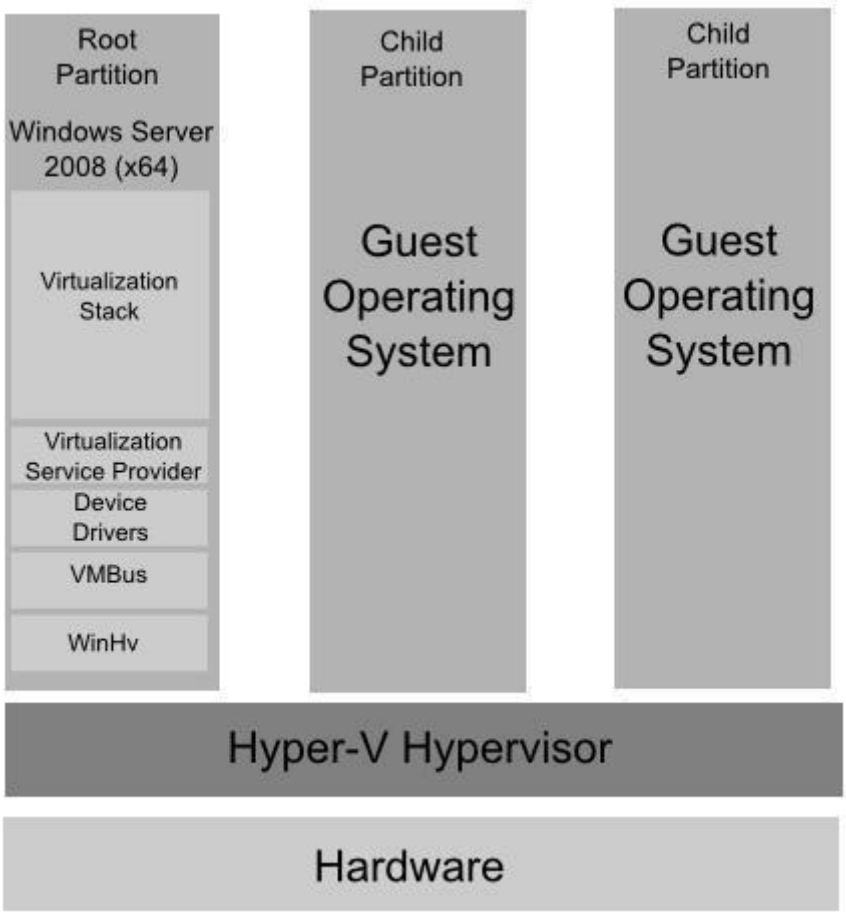


Рисунок 2.2 – Архітектура Microsoft Hyper-V [4]

Підтримувані операційні системи віртуальних машин [5]:

- Windows Server 2012 R1/R2;
- Windows Server 2008 x86/x64 SP1/SP2 и R2;
- Windows HPC Server 2008;
- Windows Server 2003 x86/x64 SP2 R2;
- Windows 2000 Server SP4 и Advanced Server SP4;
- Windows 7 и Windows 8.1 Pro и Windows 8.1 Update 1;
- Windows 10 x86-64;
- Windows Vista SP1/SP2 (кроме Home edition);
- Windows XP Professional SP2/SP3/x64;
- SUSE Linux Enterprise Server (SLES) 10 SP3 и 11;
- Red Hat Enterprise Linux (RHEL) 5.2 – 5.6 (x86 Edition или x86-64 Edition);

- Red Hat Enterprise Linux (RHEL) 6.0, 6.1 (x86 Edition или x86-64 Edition);
- CentOS 5.2 — 5.6, 6.0;
- FreeBSD 8.2-8.3;
- Ubuntu 12.04;
- Debian 7.6;

2.3.2 Використання гіпервізора VMware ESX Server

Даний продукт являє собою комплексне рішення для використання віртуалізації в масштабах підприємства і здійснює віртуалізацію на основі гіпервізора. Характерною відмінністю даного продукту від Microsoft Hyper-V і Xen є відсутність необхідності в окремій керуючої операційної системи – її роль виконує спеціалізоване ядро Linux, що входить до складу продукту, що робить можливим його установку на вузли, які не мають взагалі ніякої операційної системи. Таким чином, вирішується відразу цілий ряд проблем, пов'язаних з підбором і установкою керуючої операційної системи.

Vmkernel обробляє ЦП і RAM безпосередньо, використовуючи для обробки спеціальних або привілейованих інструкцій ЦП і таблиця розподілу системних ресурсів для відстеження виділеної пам'яті.

Доступ до іншого обладнання (наприклад, до мережі або пристроїв зберігання) здійснюється за допомогою модулів. Принаймні, деякі модулі є похідними від модулів, що використовуються в ядрі Linux. Для доступу до цих модулів додатковий модуль vmklinux реалізує інтерфейс модуля Linux. Згідно з файлом README, " цей модуль містить шар емуляції Linux, який використовується vmkernel

Архітектура VMware ESX Server повністю реалізовує концепцію гіпервізора. Це можна побачити на рисунку2.3 де зображена архітектура.

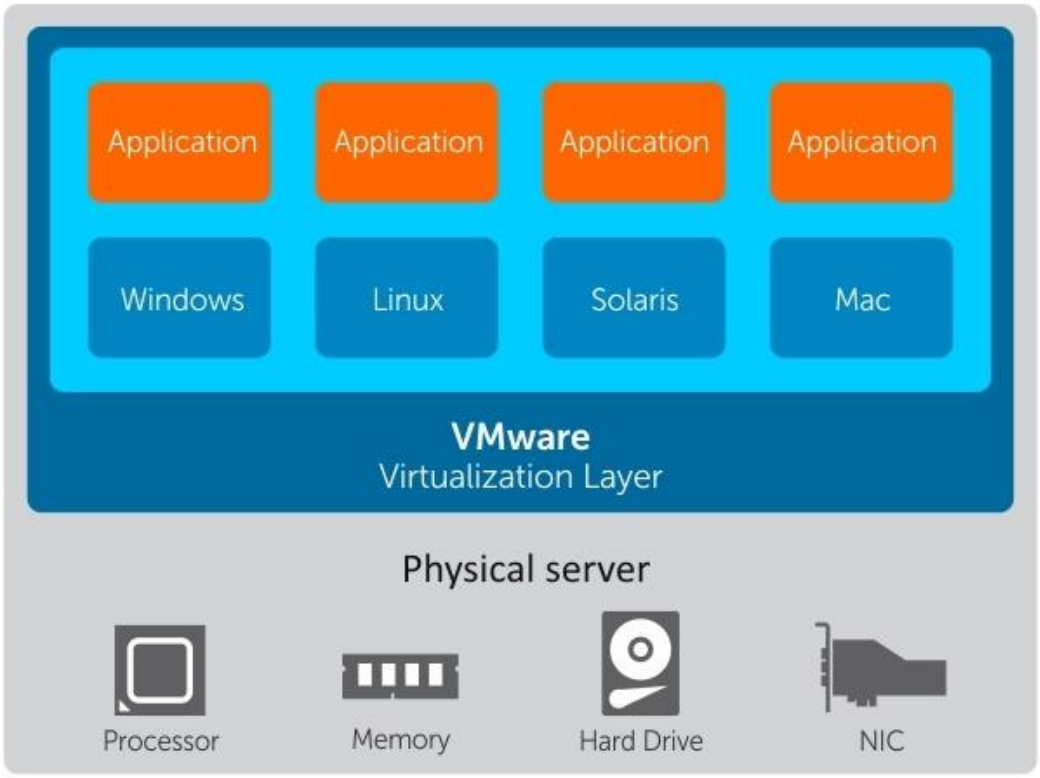


Рисунок 2.3 – Архітектура VMware ESX Server [6]

Операційні системи які підтримує VMware ESX Server [7]:

- Windows Server 2003 та вище;
- Windows 95 та вище;
- MS-DOS 6.22 and Windows 3.1x;
- FreeBSD 4.0 та вище;
- IBM OS/2 Warp 4.5.2;
- IBM OS/2 Warp 4.0;
- OS X 10.7 та вище;
- Mac OS X Server 10.5 та вище;
- Netware 4.2 Server та вище;
- Solaris 8 та вище;
- SCO OpenServer 5.0;
- SCO UnixWare 7.0;
- Asianux Server 3.0 та вище;

- CentOS 4 та вище;
- CoreOS;
- Debian 4 та вище;
- Fedora 16 Desktop Edition та вище;
- Mandrake Linux 8.x та вище;
- Mandriva Corporate 4;
- NeoKylin Linux Desktop 6.x;
- NeoKylin Linux Advanced Server 7;
- Novell Linux Desktop 9;
- openSUSE Linux 10.x та вище;
- Oracle Enterprise Linux 5;
- Oracle Enterprise Linux 4;
- VMware Photon OS;
- Red Hat Enterprise Linux Atomic Host;
- Red Hat Enterprise Linux 2.1 та вище;
- Red Hat Linux 6.2 та вище;
- Sun Java Desktop System;
- Sun Java Desktop System 2;
- SUSE Linux Enterprise Server 7 та вище;
- SUSE Linux 7.3 та вище;
- Ubuntu Linux 5.04 та вище;

Аналізуючи список підтримуваних ОС можна зробити висновок що VMware ESX Server охоплює більше різних ОС, а з цим і більше додатків які написані під різні ОС. Що неодмінно є перевагою над Microsoft Hyper-V.

2.3.3 Використання гіпервізора Xen

Xen є гіпервізором віртуальних машин і може використовуватися для створення віртуальних машин як шляхом паравіртуалізації, так і повної

віртуалізації, що дозволяє запускати в середовищі гіпервізора операційні системи з немодифікованим ядром, за умови підтримки апаратної віртуалізації.

Максимальним рівнем привілеїв виконання машинних команд має сам гіпервізор, тоді як керуюча операційна система і віртуальні машини виконуються з більш низьким рівнем привілеїв. І хоча для гіпервізора немає принципової різниці між керуючою операційною системою і системами віртуальних машин, керуюча система все ж володіє більш високим рівнем привілеїв у порівнянні з віртуальними машинами. Вона використовується для управління гіпервізором, має доступ до апаратури, надає основний інтерфейс і драйвери пристроїв. У термінології Xen будь-яка операційна система, що працює в його середовищі, називається доменом. Керуюча операційна система називається нульовим доменом (англ, domain 0) або коротко – domO. Операційні системи віртуальних машин називаються звичайними доменами (англ, domain unprivileged) або коротко – domU. З урахуванням наведених вище позначень легко уявити собі архітектуру вузла, що працює під управлінням Xen, у вигляді рисунка 2.4.

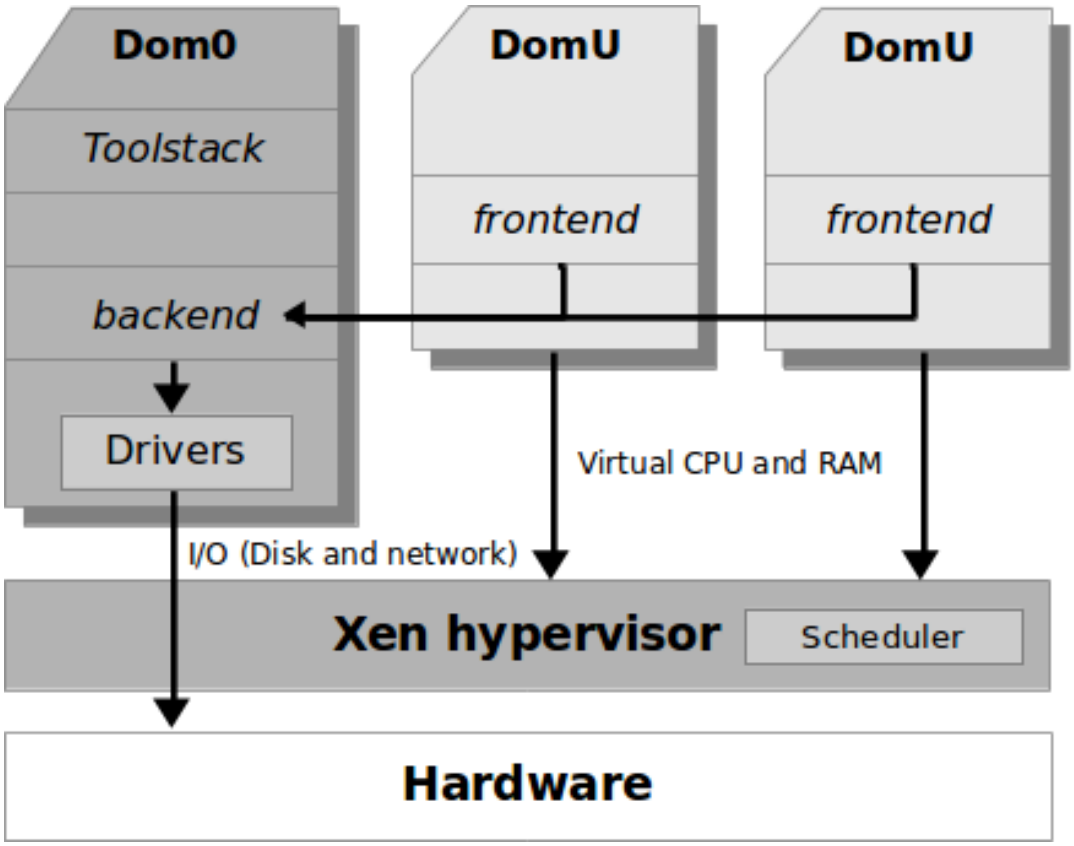


Рисунок 2.4 - Архітектура Xen вузла [8]

Підтримувані операційні системи [9]:

- Linux Debian GNU;
- Linux SuSE;
- Linux Fedora;
- OpenSolaris;
- NetBSD;
- FreeBSD;
- OpenBSD;
- Windows Server 2003;
- Windows Server 2008;
- Windows XP;
- Windows Vista;
- Windows 7.

2.3.4 Використання гіпервізора KVM

KVM (kernel virtual machine) розшифровується як віртуальна машина ядра, тобто гіпервізор включений в ядро, за рахунок чого досягається дуже висока продуктивність віртуалізації. KVM є відкритим програмним забезпеченням, поширюється за ліцензією GPL v2 і складається з ядра, модуля взаємодії з процесором і компоненти користувальницького режиму. Для кращої продуктивності, процесор повинен підтримувати набір інструкцій для віртуалізації: Intel VT або AMD-V.

`KVM використовується як в корпоративних рішеннях (Redhat) так і для малих об'ємів віртуалізації. Вже кілька років ця технологія є основною для RHEL (Red Hat Enterprise Linux) замість XEN.

Програма, яка працює в просторі користувача, використовує інтерфейс KVM, здійснюючи через нього налаштування адресного простору гостьової віртуальної

системи, використовує її I/O ресурси, відображаючи образ гостьової системи на образ хоста. З архітектурою KVM ви можете ознайомитися на рисунку 2.5.

Процес kvm працює в основному як звичайна програма Linux. Він виділяє свою пам'ять за допомогою звичайних викликів malloc або mmap. KVM успадковує потужні функції управління пам'яттю від Linux. Пам'ять віртуальної машини зберігається так само, як пам'ять будь-якого іншого Linux-процесу, і може замінюватися, копіюватися великими сторінками для підвищення продуктивності, узагальнюватися, або зберігатися у файлі на диску. Підтримка технології NUMA дозволяє віртуальним машинам ефективно звертатися до пам'яті великого обсягу.
KVM підтримує новітні функції віртуалізації пам'яті від виробників процесорів, зокрема, Intel Extended Page Table (EPT) і AMD Rapid Virtualization Indexing (RVI), для мінімізації завантаження процесора і досягнення високої пропускної здатності.
Узагальнення сторінок пам'яті підтримується за допомогою функції ядра Kernel Same-page Merging. Kernel Same-page Merging сканує пам'ять кожної віртуальної машини, і якщо якісь сторінки пам'яті віртуальних машин ідентичні, об'єднує їх в одну сторінку, яка стає загальною для цих віртуальних машин і зберігається в єдиній копії. Якщо гостьова система намагається змінити цю загальну сторінку, їй надається власна копія.

Оскільки віртуальна машина реалізована як Linux-процес, вона використовує стандартну модель безпеки Linux для ізоляції та управління ресурсами. З допомогою SELinux ядро Linux додає обов'язкові засоби контролю доступу, багаторівневі і різноманітні засоби захисту, а також управляє політикою безпеки. SELinux забезпечує сувору ізоляцію ресурсів і обмежує рухливість процесів, запущених в ядрі Linux.

Компонент SVirt інтегрує функції безпеки використовуючи технологію мандатного управління доступом і віртуалізацію на базі Linux (KVM). Svirt побудований на SELinux, щоб забезпечити інфраструктуру, яка дозволить адміністратору визначати політику ізоляції віртуальних машин. SVirt покликаний гарантувати, що ресурси віртуальних машин не будуть доступні для жодних інших процесів, або віртуальних машин. Адміністратор може доповнити цю політику,

визначивши детальні Політики дозволу. Наприклад, щоб група віртуальних машин спільно використовувала одні і ті ж ресурси.

KVM може використовувати будь-який носій, підтримуваний Linux, для зберігання образів віртуальних машин, у тому числі локальні диски з інтерфейсами IDE, SCSI, SATA, NAS, включаючи NFS і SAMBA/CIFS, або SAN з підтримкою iSCSI і Fibre Channel. Для поліпшення пропускної здатності системи зберігання даних і резервування може використовуватися багатопотоковий I/O.

Знаходячись у складі ядра Linux, KVM може використовувати перевірену і надійну інфраструктура зберігання даних з підтримкою всіх провідних виробників.

KVM підтримує образи віртуальних машин в розподілених файлових системах, таких як Global File System, так що вони можуть поділятись на кілька хостів або узагальнюватися з використанням логічних томів. Підтримка тонкої налаштування образів дисків дозволяє оптимізувати використання ресурсів зберігання даних, виділяючи їх не відразу, а тільки тоді, коли цього вимагає віртуальна машина. Власний формат дисків для KVM — QCOW2 забезпечує підтримку знімків поточного стану та забезпечує кілька рівнів таких знімків, а також стиснення і шифрування.

KVM підтримує концепцію живу міграцію, забезпечуючи можливість переміщення працюючих віртуальних машин між фізичними вузлами без переривання обслуговування. Жива міграція прозора для користувачів. Віртуальна машина залишається включеною, мережеві з'єднання активними, і додатки користувача продовжують працювати, в той час як віртуальна машина переміщається на новий фізичний сервер. Крім концепції живої міграції, KVM підтримує концепцію холодної міграції, дозволяючи зберігати і відновлювати її пізніше.

KVM підтримує гібридну віртуалізацію, коли паравіртуалізовані встановлені драйвери у гостьовій операційній системі дозволяють віртуальним машинам використовувати оптимізований інтерфейс вводу/виводу, а не емулюють пристрій, забезпечуючи високу продуктивність введення/виводу для мережевих і блокових пристроїв.

Гіпервізор KVM використовує стандарт VirtIO, розроблений IBM і Red Hat спільно з Linux-співтовариством для паравіртуалізованних драйверів. VirtIO — це незалежний від гіпервізора інтерфейс для створення драйверів пристроїв, що дозволяє декільком гіпервізорам використовувати один і той же набір драйверів пристроїв, що покращує взаємодію між гостьовими системами.

Драйвери VirtIO входять в сучасні версії Linux-ядра починаючи з версії 2.6.25, включені в Red Hat Enterprise Linux починаючи з версії 4.8. Red Hat розробила драйвери VirtIO для гостьових ОС Microsoft Windows, що оптимізують і мережеві дискові операції введення/виводу. Ці драйвери сертифіковані за програмою сертифікації Microsoft Windows Hardware Quality Labs.

Управління емуляцією пристроїв здійснюється за допомогою модифікованої версії qemu, забезпечує емуляцію BIOS і всіх стандартних шин, а також набір системних пристроїв (контролери IDE і SCSI, мережні карти і т. д.).

KVM успадкував продуктивність і масштабованість Linux, підтримуючи віртуальні машини з 16 віртуальними процесорами і 256 ГБ віртуальної пам'яті кожна, а також хост-системи з 256 ядрами і 1 ТБ RAM.

Він може забезпечити:

- продуктивність в 95-135% порівняно з "голим залізом" в реальних корпоративних додатках, таких як SAP, Oracle, LAMP і Microsoft Exchange;
- понад мільйон повідомлень в секунду і менш ніж 200 мкс затримку у віртуальних машинах, що працюють на стандартному сервері;
- максимальні рівні консолідації більш ніж з 600 віртуальними машинами, що виконують корпоративні додатки, на одному сервері.

Це означає, що KVM допускає віртуалізацію найвимогливіших робочих навантажень.

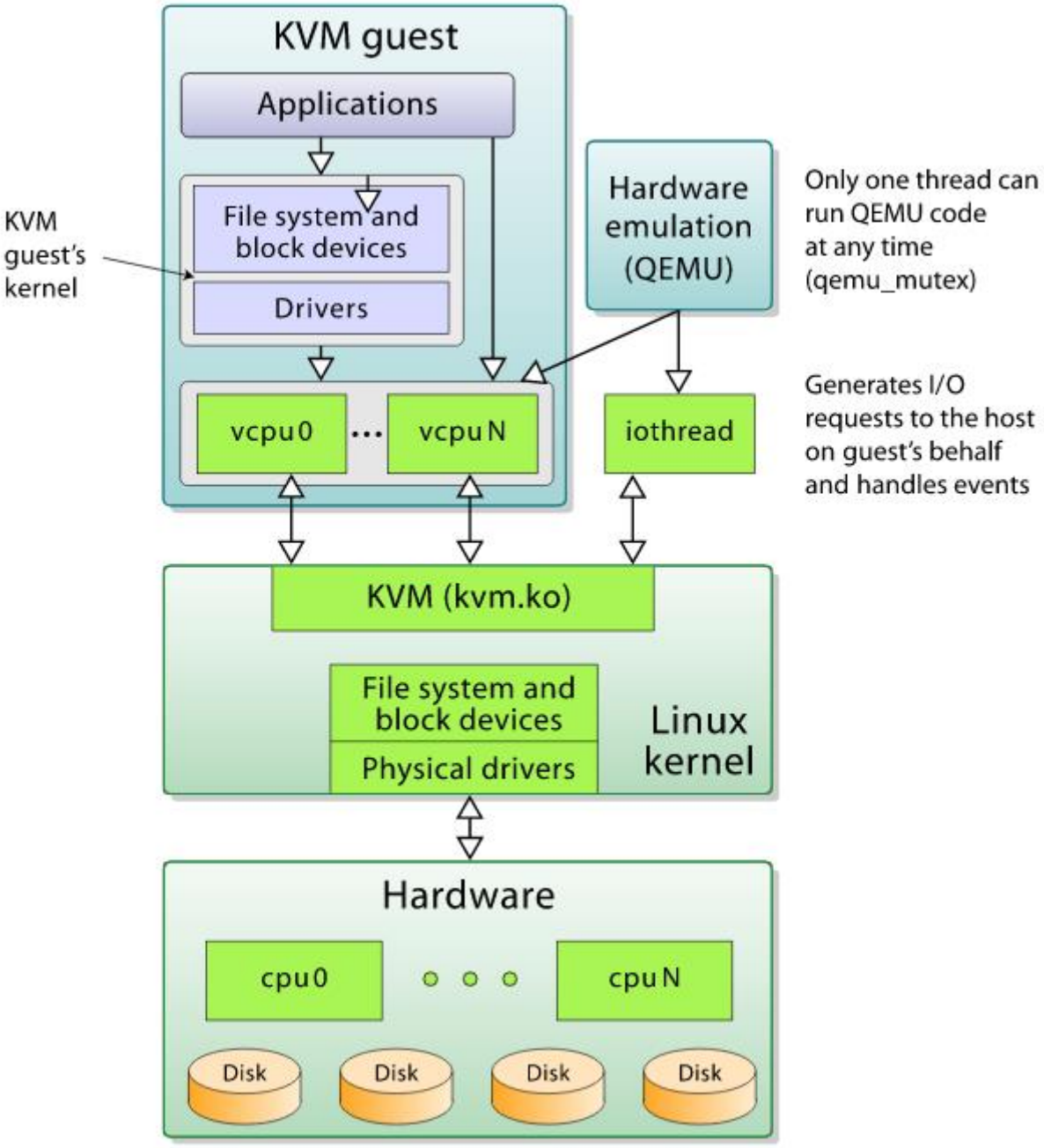


Рисунок 2.5 – Архітектура KVM [10]

Підтримувані операційні системи [11]:

– Windows Server 2003 та вище;
– Windows XP та вище;
– CentOS 5.5 та вище;
– Fedora 4 та вище;
– RedHat Enterprise Linux 8 та вище;
– Debian 4 та вище;

- Ubuntu 8 та вище;
- Xandros 3 OCE;
- Knoppix 6.7.1 та вище;
- Android 2.2;
- SUSE Linux Enterprise Server 10;
- SUSE Linux Enterprise Server 11;
- openSUSE 10.3 та вище;
- Gentoo;
- Arch Linux;
- OpenBSD 4.0 та вище;
- FreeBSD 6.0 та вище;
- NetBSD 4.1 та вище;
- Solaris 10 та вище;
- MS DOS 5.0 та вище.

2.4 Використання віртуальної машини

Віртуальна машина – це контейнер, який має різний формат компонентів та різну взаємодію з гіпервізорами. Але не зважаючи на це, можна виділити чотири основні компоненти які має віртуальна машина:

- ядро операційної системи;
- драйвери;
- систему вводу/виводу (I/O);
- додатки користувача.

Ядро операційної системи може функціонувати по різному в залежності від того, чи є підтримка апаратної віртуалізації на фізичному сервері. Також від того яку архітектуру має ядро ОС і чи відрізняється вона від архітектури CPU фізичного сервера. У випадку якщо архітектура ядра відрізняється від архітектури CPU фізичного сервера на якому має функціонувати ВМ, то сама віртуальна машина перетворюється на емулятор. Емулятор взагалі може не мати гіпервізора як окремої

структури, він може бути вмонтований в сам контейнер ВМ. Емулятор, зазвичай, не може бути встановлений на фізичний сервер на пряму, тому він встановлюється на операційну систему і функціонує звертаючись до ОС. Емулятори не є ціллю системи управління тому більш детально розглядатись у цій дипломній роботі вони не будуть.

Якщо архітектура ядра ОС і архітектура CPU на фізичному сервері співпадають, але фізичний сервер не підтримує апаратну віртуалізацію, гіпервізор надає в користування ВМ конкретну кількість ресурсів, так ніби ядро ОС функціонує самостійно на фізичному сервері. При звернені ядро ОС до ресурсу CPU, гіпервізор оброблює команди, направляє команди до CPU в обробленому вигляді, отримує результат від CPU, обробляє результат та повертає оброблений результат до ядра ОС.

Якщо архітектура ядра ОС і архітектура CPU на фізичному сервері співпадають, фізичний сервер та ядро ОС підтримує апаратну віртуалізацію, гіпервізор використовує технологію черг для команд CPU, та інший спосіб надання простору RAM. Технологія черг для команд CPU відрізняється від попередньо оглянутої технології тим, що не обробляє команди які йдуть від ядра ОС до CPU. Замість цього команди які надходять до гіпервізора становляться у чергу до фізичних ядер CPU, разом із командами інших віртуальних машин, та командами гіпервізора. Також за допомогою технології віртуалізації RAM, гіпервізор надає адреси реальних сторінок пам’яті віртуальний машині лише контролюючи адресацію з метою запобігання конфлікту між Віртуальними машинами які працюють на одному фізичному сервері.

Доступ до зовнішніх пристроїв, які використовуються віртуальною машиною, також може бути як прямий, так і віртуалізований. У випадку відсутності підтримки віртуалізації зовнішніх пристроїв, гіпревізор самостійно направляє команди від ВМ. Коли віртуалізація зовнішніх пристроїв підтримується, драйвери віртуальної машини надсилають команди не до гіпервізора, а до зовнішніх пристроїв напряму.

Комунікація між компонентами віртуальної машини, гіпервізора, апаратного обладнання неможливе без інтерфейсу вводу/виводу (I/O). I/O створює необхідну

взаємодію між усіма компонентами системи. Саме через цей інтерфейс компоненти системи мають змогу надсилати та отримувати як команди, так і дані для обробки.

2.5 Використання балансувальника навантаження

У зв'язку з масовим поширенням розподілених обчислювальних систем стала актуальною проблема їх ефективного використання. Одним з аспектів даної проблеми є ефективне планування і розподіл завдань всередині розподілених обчислювальних систем з метою оптимізації використання ресурсів і скорочення часу обчислення. Досить часто виникає ситуація, при якій частина обчислювальних ресурсів простоює, в той час, як інша частина ресурсів перевантажена і в черзі є велика кількість завдань які очікують свого виконання завдань.

Для оптимізації використання ресурсів, скорочення часу обслуговування запитів, горизонтального масштабування, динамічне додавання, видалення пристроїв, а також забезпечення відмовостійкості за рахунок резервування застосовується метод рівномірного розподілу завдань між кількома мережевими фізичними серверами так званий балансування навантаження.

Проблема балансування обчислювального навантаження розподіленої системи виникає по наступних причин:

– структура обчислювального комплексу (наприклад, кластера) неоднорідна, тобто різні обчислювальні вузли мають різну продуктивність;

– структура межузлового взаємодії неоднорідна, тому що лінії зв'язку, що з'єднують вузли, можуть мати різні характеристики пропускної здатності;

– структура розподіленого додатка неоднорідна, різні логічні процеси вимагають різних обчислювальних потужностей.

Експериментальні та чисельні дослідження, проведені в останні десятиліття, свідчать, що трафік в комп'ютерних мережах має самоподібні властивості. Самоподібний трафік викликає значні затримки і втрати пакетів, навіть якщо сумарна інтенсивність всіх потоків далека від максимально допустимих значень. Самоподібні властивості інформаційних потоків виявлені в багатьох локальних і

глобальних телекомунікаційних мережах. У зв'язку з вищезгаданим, почали активно досліджуватися механізми підвищення якості обслуговування і методів управління трафіком в комп'ютерних мережах, які функціонують в умовах самоподібного і мультифрактального трафіку.

При появі нових завдань програмне забезпечення, що реалізує балансування, має прийняти рішення про те, на якому обчислювальному вузлі слід виконувати обчислення, пов'язані з цим новим завданням. Крім того, балансування передбачає перенесення частини обчислень з найбільш завантажених обчислювальних вузлів на менш завантажені вузли. При виконання завдань процесори обмінюються між собою комунікаційними повідомленнями. В у разі низьких витрат на комунікацію, деякі процесори можуть простоювати, в той час як решта будуть перевантажені. Також будуть недоцільні великі витрати на комунікацію між вузлами. Отже, стратегія балансування повинна бути такою, щоб обчислювальні вузли були завантажені досить рівномірно, але і комунікаційне середовище не повинно бути перевантажене.

Алгоритми балансування навантаження можуть бути класифіковані за кількома типами.

При використанні статичного алгоритму поточний стан вузла не враховується, тому потрібна попередня баз знань про статистику кожного вузла і користувальницьких вимогах. Статичний алгоритм не гнучкий і погано маштабовується.

Динамічний алгоритм працює відповідно до динамічних змін стану вузлів, тобто він збирає, зберігає і аналізує інформацію про стан системи. Тому виникають великі накладні витрати на балансування. Необхідно враховувати розташування процесора, якому передається навантаження від перевантаженого процесора, оцінку навантаження, обмеження числа міграцій. Якщо який-небудь вузол дав збій, то це не зупинить роботу всієї мережі, але вплине на продуктивність системи.

Динамічний підхід балансування навантаження підрозділяється на два типи: розподілений і нерозподілений підходи. Вони визначається наступним чином.

У нерозподіленому підході один вузол або група вузлів відповідають за управління і розподіл по всій системі. Інші вузли не розподіляють завдання і не виконують керуючі функції, отже цей тип алгоритмів не відмовостійкий, може відбутися перевантаження центрального процесора. Корисні в невеликих мережах з низьким навантаженням.

В розподіленому підході кожен вузол незалежно будує свій вектор навантаження. Всі процесори в мережі відповідальні за розподіл навантаження і наповнення власної локальної бази даних для прийняття ефективних рішень балансування. Це призводить до великих комунікаційних витрат і складності алгоритмів. Корисні в великих і гетерогенних мережах.

При розподіленому підході балансування навантаження може мати дві форми: кооперативну і некооперативную. У кооперативній вузли працюють разом для досягнення спільної мети. Наприклад, поліпшення загального часу відгуку. При некооперативному підході балансування вузол працює незалежно від мети, наприклад, поліпшення часу виконання місцевого завдання. Вузли постійно взаємодіють один з одним і при розподіленому підході генерують більше повідомлень, ніж при нерозподіленому. Передача повідомлень між вузлами для обміну інформацією про оновлення системи може призвести до зниження продуктивності системи. Один вузол або група вузлів вирішує завдання балансування навантаження при нерозподіленому підході, він може приймати дві форми централізованої і напіврозподіленої.

У централізованих алгоритмах один вузол несе виняткову відповідальність за балансування навантаження всієї системи і називається центральним вузлом. Використовується в невеликих мережах. У напіврозподілених алгоритмах кластер формується групою вузлів системи. Балансування навантаження відбувається в кожному кластері за централізованим типом. Серед вузлів в кластері центральний вузол ініціалізує балансування навантаження в цій групі.

Напіврозподілені алгоритми обмінюються великою кількістю повідомлень у порівнянні з централізованими. Зазвичай динамічний алгоритм балансування містить чотири елементи:

– Політику балансування;
– алгоритм вибору партнера;
– алгоритм вибору завдання для передачі;
– механізм збору необхідної інформації про стан системи.

Політика балансування визначає, чи є вузол об'єктом балансування Різні види політик балансування використовують порогові значення завантаженості вузлів, відхилення величини навантаження вузла від середнього значення по системі та інші методи. Алгоритм вибору завдання визначає, яке завдання необхідно передати. При виборі завдання для відправки алгоритм може враховувати, що накладні витрати, пов'язані з пересиланням завдання, повинні бути мінімальні, складність виконання завдання повинна бути великою, число зв'язків в задачі з локальними ресурсами має бути мінімально. Алгоритм вибору партнера відповідає за вибір відповідного вузла для операції балансування. Поширеною технікою в розподілених алгоритмах є опитування (polling) вузлів. Опитування може бути послідовним або паралельним, використовувати результати попередніх опитувань. В централізованих алгоритмах вузол звертається до спеціалізованого координатору для визначення відповідного партнера для балансування. Координатор збирає і підтримує в актуальному стані інформацію про завантаженість вузлів системи, а алгоритм пошуку партнера використовує ці дані.

Механізм збору інформації про завантаженість системи визначає джерело інформації, час збору даних про завантаженість, місце зберігання інформації.

Існує кілька класів механізмів збору інформації:

– Збір даних по необхідності. В даному класі використовуються розподілені алгоритми, які збирають інформацію про завантаженість, коли вузол потребує балансування навантаження.

– Періодичний збір даних. Алгоритми даного класу можуть бути як централізованими, так і розподіленими. Залежно від зібраних даних, алгоритм ініціює балансування навантаження.

– Збір даних по зміні стану. В системах, що реалізують алгоритми даного класу, вузли самі поширюють інформацію про зміну завантаженості при зміні

внутрішнього стану. У разі розподілених алгоритмів дані направляються сусіднім вузлам, у разі централізованих алгоритмів дані направляються координатору.

Системи балансування навантажень діляться на категорії:

– апаратні пристрої;
– мережеві комутатори;
– програмні рішення.

Системи балансування на базі апаратного пристрою зазвичай функціонують під управлінням UNIX або фірмової ОС, на якій встановлена розроблена постачальником система балансування навантаження. Такі системи відповідають специфікації Plug and Play (PnP), що полегшує роботу адміністраторів вузлів. Для реалізації систем балансування на базі мережевих комутаторів використовуються комутатори другого і третього рівня. На відміну від апаратних рішень ці системи не передбачають установки додаткових пристроїв, через які сервіси користувача підключаються до комутатора. Якщо при організації служби розподілу навантаження в пулі використовуються програмні продукти, то можна обійтися без модифікації наявних мережевих засобів і обладнання. Програмні пакети встановлюються на існуючих серверах або на спеціальних серверах вирівнювання навантаження.

Незалежно від того, до якої категорії відноситься та чи інша система балансування навантаження, вона виконує наступні завдання:

– контроль за навантаженням і станом серверів;
– правильний вибір сервера;
– управління трафіком між клієнтом і сервером.

2.5.1 Моніторинг стану серверів

Система балансування навантаження постійно відстежує рівень навантаження і стан ввірених їй серверів з тим, щоб на підставі зібраної інформації в будь-який момент надати користувачу доступ до того сервера, який зможе найкращим чином

відповісти на його запит. При цьому використовується два методи контролю: зовнішній моніторинг і внутрішній моніторинг.

При проведенні зовнішнього моніторингу система балансування навантаження розраховує час відгуку сервера, для чого надсилає на сервер запит і заміряє час відповіді. Найпростіша техніка виконання зовнішнього моніторингу сервера передбачає використання ping-команд по протоколу керування повідомленнями Internet Control Message Protocol (ICMP). Ці тести дозволяють системі переконатися в готовності сервера до роботи і дізнатися, скільки часу необхідно для передачі інформації з сервера на систему балансування і назад. Якщо система балансування навантаження не отримує відповіді від сервера після декількох послідовних запитів, вважається, що даний сервер недоступний. Як правило, адміністратори підключають сервери безпосередньо до системи балансування навантаження, тому якщо час, що витрачається на передачу, занадто велике, система підсумовує, що сервер працює з великим навантаженням.

Тут, однак, треба відзначити, що в ході ping-команд сервера по протоколу ICMP діагностується тільки стек протоколів IP. Описаний метод не дає уявлення про стан стека TCP. Щоб переконатися в правильності функціонування стека TCP сервера, засіб балансування навантаження робить спробу встановити з'єднання по протоколу TCP, для чого потрібно здійснити складається з трьох етапів обмін підтверджують повідомленнями. Робиться це так. Спочатку засіб балансування навантаження направляє сервера TCP-пакет, в якому значення біта syn встановлено рівним 1. Якщо після цього система балансування отримує від сервера TCP-пакет, в якому значення біта SYN дорівнює 1, а значення біта ACK теж встановлено рівним 1, вона направляє серверу другий TCP-пакет зі значенням біта SYN рівним 0 і значенням біта ACK рівним 1. Якщо обмін підтверджуючими повідомленнями завершився успішно, значить, TCP-стек сервера функціонує нормально. По завершенні такого обміну засіб балансування навантаження негайно розриває з'єднання з сервером, щоб виключити непродуктивний використання його ресурсів. Якість TCP-з'єднання з сервером оцінюється системою за таким показником, як час, необхідний для виконання всіх трьох етапів обміну підтверджують повідомленнями.

Поряд з тестуванням стеків протоколів кращі засоби балансування навантаження можуть забезпечувати моніторинг часу відгуку і готовності як самого додатка користувача, так і встановлених на ньому програм ще одним способом. Наприклад, на сервер направляється запит по протоколу HTTP на отримання інформаційних матеріалів або адреси URL. Система балансування навантаження може ініціювати передбачену по протоколу HTTP команду Get, запрошуючи тим самим у сервера вміст сторінки index.htm. Якщо код повернення, що направляється системі Web-сервером буде 200, значить початкова сторінка на фізичному сервері недоступна. Час відгуку визначається системою балансування навантаження як час з моменту відправки запиту на надання інформації до моменту отримання коду повернення.

Однак при тому, що зовнішній моніторинг дає можливість отримати про сервер корисну інформацію, як метод діагностики він має свої недоліки. Про таких суттєвих характеристиках сервера, як стан центрального процесора, пам'яті, системної шини, шини вводу/виводу, мережевої інтерфейсної плати, а також про ряд важливих ресурсів системи і прикладних програм адміністратор має лише уривчасті відомості або взагалі ніяких. Детальну інформацію про навантаження сервера може надати тільки внутрішній моніторинг. Для його виконання в системі балансування навантаження передбачені спеціальні агенти внутрішнього моніторингу, які встановлюються на кожному сервері. Агент постійно контролює стан свого "середовища проживання" і повідомляє про нього засобу балансування. Багато постачальники пропонують інструментальні засоби для роботи зі сценаріями, які дозволяють адміністраторам створювати утиліти внутрішнього моніторингу для Web-додатків. Внутрішній моніторинг широко застосовується в програмних системах балансування, але в апаратних пристроях і в рішеннях на базі комутаторів цей метод діагностики реалізується рідко.

2.5.2 Спосіб вибіру сервера

З урахуванням інформації, отриманої в ході зовнішнього і внутрішнього моніторингу серверів, система балансування навантаження може виділити сервер, який здатний краще за інших впорається з обробкою клієнтського запиту. Якщо робочі характеристики апаратних і програмних компонентів всіх серверів пулу рівнозначні, можна налаштувати систему балансування навантаження так, щоб вона виділяла сервер для обробки чергового запиту за принципом кругового списку, враховуючи при цьому стан сервера. Але якщо у IT-інфраструктурі існують сервери з різним півнем швидкодії, існує можливість відрегулювати цю систему для роботи в іншому режимі. Більш потужного серверу буде діставатися більше число запитів за рахунок вагових коефіцієнтів.

Системи балансування навантаження дозволяють адміністратору визначати правила вибору сервера на свій розсуд. Можна, наприклад, включити в ці правила такі критерії, як коефіцієнт завантаження ЦП і пам'яті, число відкритих з'єднань TCP і кількість пакетів, що надходять на мережеву інтерфейсну плату того чи іншого сервера.

У деяких випадках після того, як засіб балансування призначає сервер для виконання запиту клієнта і клієнт встановлює початкове з'єднання, для забезпечення виконання прикладної програми необхідно, щоб система балансування постійно підтримувала трафік між клієнтом і сервером. Таке з'єднання називається постійним. Якщо сервер обробляє запит клієнта, кешує інформацію про вміст додатка користувача, система балансування навантаження вже не може виділити для наступних обмінів з іншим додатком інший сервер, навіть якщо того вимагає співвідношення навантажень серверів. При такій переадресації дані про вміст віртуального кошика клієнта будуть втрачені, бо новий сервер не має цих відомостей. Система балансування повинна фіксувати інформацію про те, з яким сервером зв'язується кожен клієнт, і зберігати її в пам'яті протягом певного часу (час зберігання вказує адміністратор з урахуванням таких факторів, як звичайна поведінка клієнтів і особливості прикладної програми). При активізації функції постійних (сталих) з'єднань вона буде весь час скасовувати інші правила балансування навантаження.

Єдиною точкою агрегації трафіку виступає система зберігання даних, що забезпечує обробку потоку запитів, що надійшли від споживачів мультимедійних освітніх послуг. Отже, ефективність роботи всієї системи дистанційного навчання, а так ж якість надаваних послуг безпосередньо залежить від продуктивності сховища даних. Тому для ефективного управління потоком запитів нами розроблена модель доступу до мультимедійних даних сховища хмарної системи. Ключовою відмінністю сховищ мультимедійних даних є неоднорідність розміщуваної інформації. Наприклад текстові, аудіо або відео дані і, як наслідок, різні підходи до організації доступу до них. Крім методів доступу до даних істотним є інтенсивність звернення до тих чи інших елементам, яка може бути отримана з використанням внутрішньо системних алгоритмів ідентифікації користувачів, що в свою чергу дозволяє оцінити затребуваність і спрогнозувати навантаження на пристрої системи зберігання. У зв'язку з цим важливим аспектом управління ресурсами системи, при значному збільшенні кількості одночасних запитів, є грамотна організація процесу розміщення і розподіл елементів даних по пристроям . Відмінною характеристикою хмарних сховищ є реконфігурація їх структури в залежно від споживаних ресурсів. Це в свою чергу дозволяє впроваджувати алгоритми оптимізації в плані розміщення даних всередині дискового простору, а також управляти зміною кількості використовуваних системою пристроїв. При цьому процес оптимізації розміщення не повинен призводити до зниження якості обслуговування клієнтів система зберігання даних, для чого в алгоритмах необхідно враховувати пропускну здатність мережі і максимальний обсяг даних, який можна передавати в один момент часу . Крім того необхідно враховувати поточне завантаження самих пристроїв, а також їх розташування щодо один одного і клієнтів, що підключаються до них.

2.5.3 Використання алгоритму безпосереднього зв'язування.

Відповідно до цього методу системи балансування навантаження підбирають сервер для віртуальної машини і направляють запит на нього в той самий момент, коли система отримує від гіпервізора пакет TCP SYN. При цьому система

балансування вибирає сервер, керуючись заданими правилами розподілу навантаження серверів, а також IP-адресою, що міститься в отриманому від гіпервізора пакет TCP SYN. Цей метод забезпечує високу швидкодію, але при його застосуванні система балансування просто не встигає проаналізувати іншу інформацію: зокрема, ідентифікатор сеансу зв'язку за протоколом SSL, URL-адресу і дані прикладної програми. Щоб отримати більш докладні відомості про ВМ і точніше вибрати для нього сервер, системі балансування навантаження потрібен час для аналізу інформації рівня програми. При виборі сервера за методом відкладеного зв'язування система балансування навантаження приймає рішення про призначення сервера лише по завершенні обміну в три етапи підтверджуючими повідомленнями і після установки з'єднання між нею і користувачем. Система може враховувати вміст матеріалів, що публікуються, якщо досліджує інформацію рівня прикладної програми до вибору сервера для віртуальної машини.

2.5.4 Використання алгоритму Round Robin

Round Robin являє собою перебір по круговому циклу. Перший запит передається одному серверу, потім наступний запит передається іншому і так до досягнення останнього сервера, а потім все починається спочатку.

Найпоширенішою реалізацією цього алгоритму є метод балансування Round Robin DNS. DNS-сервер зберігає пару «ім'я хоста – IP-адреса» для кожної машини в певному домені.

У числі безсумнівних плюсів цього алгоритму слід назвати, по-перше, незалежність від протоколу високого рівня. Для роботи за алгоритмом Round Robin використовується будь-який протокол, в якому звернення до сервера йде по імені.

Балансування на основі алгоритму Round Robin не потребує великої кількості обчислювальних ресурсів Використання алгоритму Round Robin не вимагає зв'язку між серверами, тому він може використовуватися як для локальних, так і для глобальної балансування. Рішення на базі алгоритму Round Robin відрізняються

низькою вартістю. Щоб вони почали працювати, досить модифікувати кілька записів в DNS.

Алгоритм Round Robin має і цілий ряд істотних недоліків. Щоб розподіл навантаження за цим алгоритмом був ефективним, потрібно щоб у кожного сервера був однаковий набір ресурсів. При виконанні всіх операцій також має бути задіяна однакова кількість ресурсів. У реальній практиці ці умови в більшості випадків виявляються нездійсненними.

Також при балансуванні за алгоритмом Round Robin абсолютно не враховується завантаженість того чи іншого фізичного сервера в складі пулу серверів. Якщо один з вузлів завантажений на 100%, в той час як інші лише на 10 – 15%, перевантажений вузол все одно буде отримувати запити. Ні про яку ефективність і в такому випадку не може бути й мови.

В силу описаних вище обставин сфера застосування алгоритму Round Robin вельми обмежена.

Існує вдосконалена версія алгоритму Round Robin, яка має назву Weighted Round Robin. Суть удосконалень полягає в наступному: кожному серверу присвоюється ваговий коефіцієнт відповідно до його продуктивністю і потужністю. Це допомагає розподіляти навантаження більш гнучко. Фізичні сервери з більшим ваговим коефіцієнтом обробляють більше запитів. Однак всіх проблем з відмовостійкістю це аж ніяк не вирішує. Більш ефективне балансування забезпечують інші методи, в яких при плануванні і розподілі навантаження враховується більша кількість параметрів.

2.5.4 Переадресування трафіку

Системи балансування навантаження можуть перенаправляти трафік клієнтів на обраний сервер кількома способами: методом трансляції адрес з керуванням доступу до середовища передачі з використанням MAC-адрес, за методом трансляції мережевих адрес NAT, при використанні зв'язування відкладеного - з допомогою механізму шлюзу TCP.

Метод трансляції адрес з керуванням доступу до середовища передачі з використанням MAC-адрес може бути реалізований системою балансування навантаження при тому умови, що кожен сервер поряд зі своїм фізичним IP-адресою використовує в якості інтерфейса адреси зворотного зв'язку пріоритетну адресу системи балансування. Отримавши від клієнта пакет і призначивши йому відповідний сервер, система балансування замінює в цьому пакеті MAC-адресу одержувача MAC-адресою відповідного сервера, після чого направляє пакет на виділений сервер. У пакеті міститься IP-адреса клієнта, так що для прямої відповіді клієнту сервер використовує в якості IP-адреси одержувача початковий IP-адресу клієнта. Однак в якості IP-адреси відправника сервер вказує пріоритетну адресу системи балансування навантаження, як якщо б трафік надходив клієнту саме від неї. Таким чином, наступний пакет від клієнта направляється не відповів йому серверу, а системі балансування навантаження.

При використанні методу трансляції мережевих адрес NAT система балансування направляє отриманий від клієнта пакет призначеному сервера лише після того, як виконає над пакетом кілька операцій. По-перше, вона заміщує в пакеті адресу власного IP -адреси IP-адресою призначеного сервера, а по-друге, змінює IP-адресу відправника на свою пріоритетну адресу. Даний метод дозволяє приховувати від клієнтів IP-адреси серверів, так що останні можуть використовувати будь-які IP-адреси, у тому числі і приватні. При цьому сервери не обов'язково повинні бути безпосередньо з'єднані з системою балансування. Достатньо мати змогу встановлювати з'єднання по протоколах статичної або мережевої маршрутизації.

При встановленні безпосереднього зв'язування системи балансування навантаження можуть направляти трафік за методами трансляції адрес з керуванням доступу або NAT на рівнях 2 або 3. Але якщо мова йде про відкладений зв'язування, системи балансування повинні управляти трафіком на рівні TCP і на більш високих рівнях. Відкладене зв'язування передбачає, що система балансування навантаження і клієнт встановлюють з'єднання по протоколу TCP так, щоб система могла отримати дані програми ще до призначення сервера. Потім засіб балансування встановлює TCP з'єднання з призначеним сервером і передає йому клієнтський запит. Далі

система балансування передає клієнту відповідь сервера, для чого використовується TCP з'єднання "система балансування - клієнт". Описана функція і називається шлюзом TCP. Її можна реалізувати в системі балансування навантаження з допомогою спеціального агента. Цей агент встановлюється на сервері, який забезпечує пряме TCP-з'єднання між клієнтом і сервером, виступаючим в ролі засоби балансування.

На основі моделі доступу до даних сховища існує рішення балансування навантаження між пристроями, реалізований у вигляді програмного модуля для компонента Swift хмарної системи OpenStack. Вибір даної хмарної системи обумовлений відкритістю її архітектури і можливістю її модифікації під поставлені завдання. Основними недоліками OpenStack є неефективний алгоритм розподілу обчислювальних завдань між вузлами зберігання даних.

Стандартний алгоритм, запропонований в системі, не враховує маршрутизацію віртуальної і топологію локальної мережі, а також віддаленість віртуальних машин, що виконують обробку запитів користувачів, і сховищ даних, забезпечують передачу даних. Все це негативно впливає на час відгуку, як самої хмарної системи, так і запущених в ній примірників додатків. Крім того, самі алгоритми розподілу даних, застосовувані в сховище хмарної системи, не дозволяють ефективно здійснювати розміщення інформації та надавати доступ до затребуваних даних по мережі.

Пропонується рішення, який дозволяє знизити час відгуку, використовуючи інформацію про топології та маршрутизації потоків даних, а гнучке управління їх розміщенням дозволяє скоротити накладні витрати обчислювальних потужностей при міграції даних і віртуальних машин.

Для оцінки ефективності розробленого алгоритму проведено моделювання роботи системи зберігання з різними параметрами. При цьому отримані наступні закономірності при роботі стандартних алгоритмів хмарної системи.

При збільшенні кількості копій даних відбувається значне зниження навантаження на основних пристроях зберігання. Однак, при цьому зростає кількість задіяних пристроїв. При одночасному доступі до кількох пристроїв, що містять різний обсяг

даних, виникає дисбаланс продуктивності сховища, що призводить до відмов в обслуговуванні запитів користувача.

Основною причиною є нерівномірне розміщення великих і малих за обсягом даних, що в свою чергу збільшує час зайнятості пристроїв. При багаторазовому зверненні до одних і тих же даних в сховище, пристрої не в змозі обслужити запити, так як відсутній розподіл навантаження між вузлами.

2.6 Підсумок структури об’єкту системи управління

Враховуючи велику кількість типів компонентів та кількість самих компонентів пулу серверів, а також враховуючи переваги та недоліки гіпервізорів описаних у підрозділах 2.3.1 – 2.3.4 можна зробити висновок що жоден гіпервізор не спроможний вирішити всі завдання які висуває користувач корпоративної IT-інфраструктури. Використовуючи єдину концепцію віртуальних машин яка була описана у підрозділі 2.4 було розроблено окремий модуль виконання за для управління віртуальними машинами у різних гіпервізорів від різних виробників. Враховуючи можливості балансувальника навантаження описаного у підрозділі 2.5 стає можливим розподіляти навантаження рівномірно серед усіх фізичних серверів, які входять до складу єдиного пулу серверів.

3 ОГЛЯД МОЖЛИВОСТЕЙ ІСНУЮЧИХ СИСТЕМИ УПРАВЛІННЯ

3.1Огляд Hyper-V Manager

Як вже згадувалось у розділі 1, віртуальні машини не можуть функціонувати самі по собі. Їм необхідний спеціальних програмних прошарок який називається гіпервізор. Операційна система яка працює з власними компонентами як віртуальна машина не здатна управляти ресурсами які ВМ надає гіпервізор. Зрозуміло, що гіпервізрор не функціонує сам по собі, тому було розроблено спеціальне забезпечення для управління його функціоналом та можливостями та контролем за функціонуванням ВМ.

Управлінням Hyper-V займается система під назвою Hyper-V Manager. Користувацьке вікно зображене на рисунку 4.1

До її функціоналу можна віднести такі можливості:

- змінювати розмір логічного диску під час роботи ВМ;
- створювати віртуальні машини;
- видаляти віртуальні машини;
- можливість міграції віртуальних машин;
- створення контрольних точок.

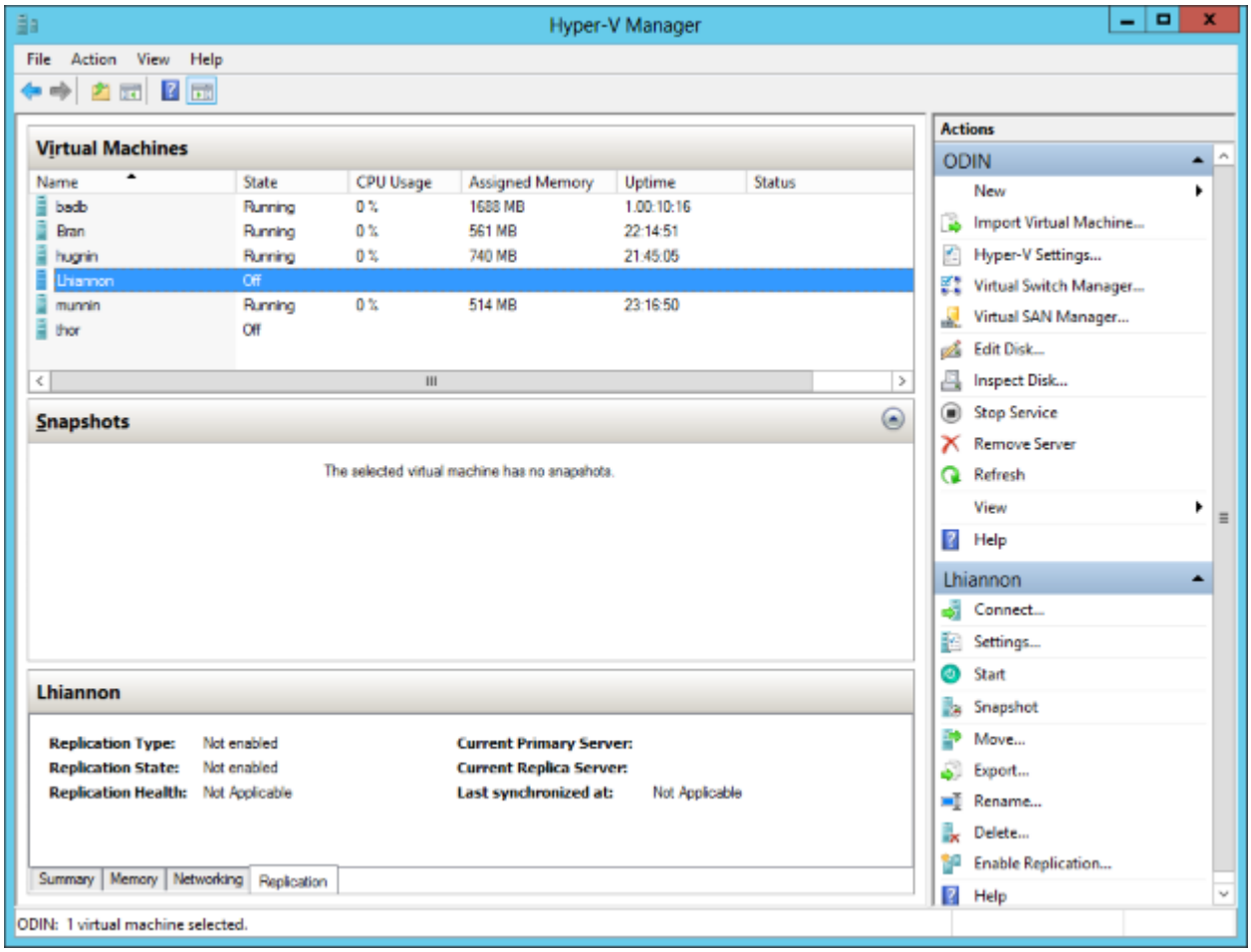


Рисунок 3.1 – Робоче вікно Hyper-V Manager [12]

3.2 Огляд VMware ESX Server

ESX Server працює безпосередньо на фізичному сервері, без участі операційної системи, на відміну від інших продуктів компанії VMware. Архітектура ядра виглядає наступним чином. Спочатку запускається ядро Linux, потім завантажуються різні спеціальні компоненти для віртуалізації, включаючи ESX. Первинним є ядро Linux, воно викликається через консоль. У нормальному режимі роботи, vmkernel працює на прямо на апаратних компонентах фізичного серверу, а сервіс консолі Linux працює, як первинна віртуальна машина.

Vmkernel це мікроядро з трьома інтерфейсами: апаратний, гостьові системи і консольний сервіс.

Кожна віртуальна машина ESX Server використовує ті ж ресурси, що і інші і вони можуть бути запущені одночасно. На відміну від інших гіпервізорів, управління ESX Server відбувається через віддалений доступ.

Основна функціональність сконцентрована в VMkernel, розмір якого, Через відсутність основної операційної системи, становить до 150 МБ. Це зменшує можливість атак для шкідливого ПЗ, збільшуючи надійність і безпеку.

Архітектура ESX Server має менше параметрів налаштування і більш зручна в розгортанні, тому віртуальну інфраструктуру на її основі простіше обслуговувати.

Для моніторингу обладнання та управління системою ESX Server використовує модель інтеграції партнерських API-інтерфейсів без агентів. Завдання управління виконуються через засоби віддаленого командного рядка vSphere Command Line Interface (vCLI) і PowerCLI, де автоматизація управління PowerCLI здійснюється за допомогою елементів cmdlet і сценаріїв Windows PowerShell.

Оптимізація доступу на основі ролей і засобів контролю виключає залежність від загального облікового запису привілейованого користувача. Користувачам і групам можна призначити повні права адміністрування. Для виконання адміністративних завдань немає необхідності мати спільний доступ, або стандартний обліковий запис адміністратора.

ESX Server веде журнали всієї активності користувачів через оболонку і інтерфейс прямої консолі. Ведення журналів забезпечує облік користувачів і спрощує аудит активності користувачів.

ESX Server забезпечує перенесення віртуальної машини цілком з одного фізичного сервера на інший без простою. Підтримується перенесення працюючих віртуальних машин по кластерах, розподілених комутаторів і серверів vCenter, а також на великі відстані (час на передачу і підтвердження до 100 мс).

Віртуальні машини, що працюють на основі ESX Server, надають такі можливості:

- Підтримка до 128 віртуальних ЦП на віртуальній машині.
- Підтримка до 4 Тбайт ОЗУ.
- Підтримка пристроїв USB 3.0 новим контролером xHCI.

– Підтримка до 120 пристроїв на віртуальну машину завдяки новому інтерфейсу Advanced Host Controller Interface (AHCI).

– Максимальний обсяг VMDK-диска — 62 Тбайт.

– Повернення дискового простору в пул ресурсів при звільненні сховища гостьової ОС.

– ESX Server передає віртуальній машині більше даних про архітектуру ЦП вузла. Це надає більш широкі можливості для налагодження, регулювання та усунення помилок операційних систем і додатків на цій віртуальній машині;

– Збільшення ефективності ЦП за рахунок підтримки технології Large Receive Offload (LRO), яка об'єднує вхідні TCP-пакети в один великий пакет.

Як саме виглядає користуватський інтерфейс можна побачити на рисунку 4.2.

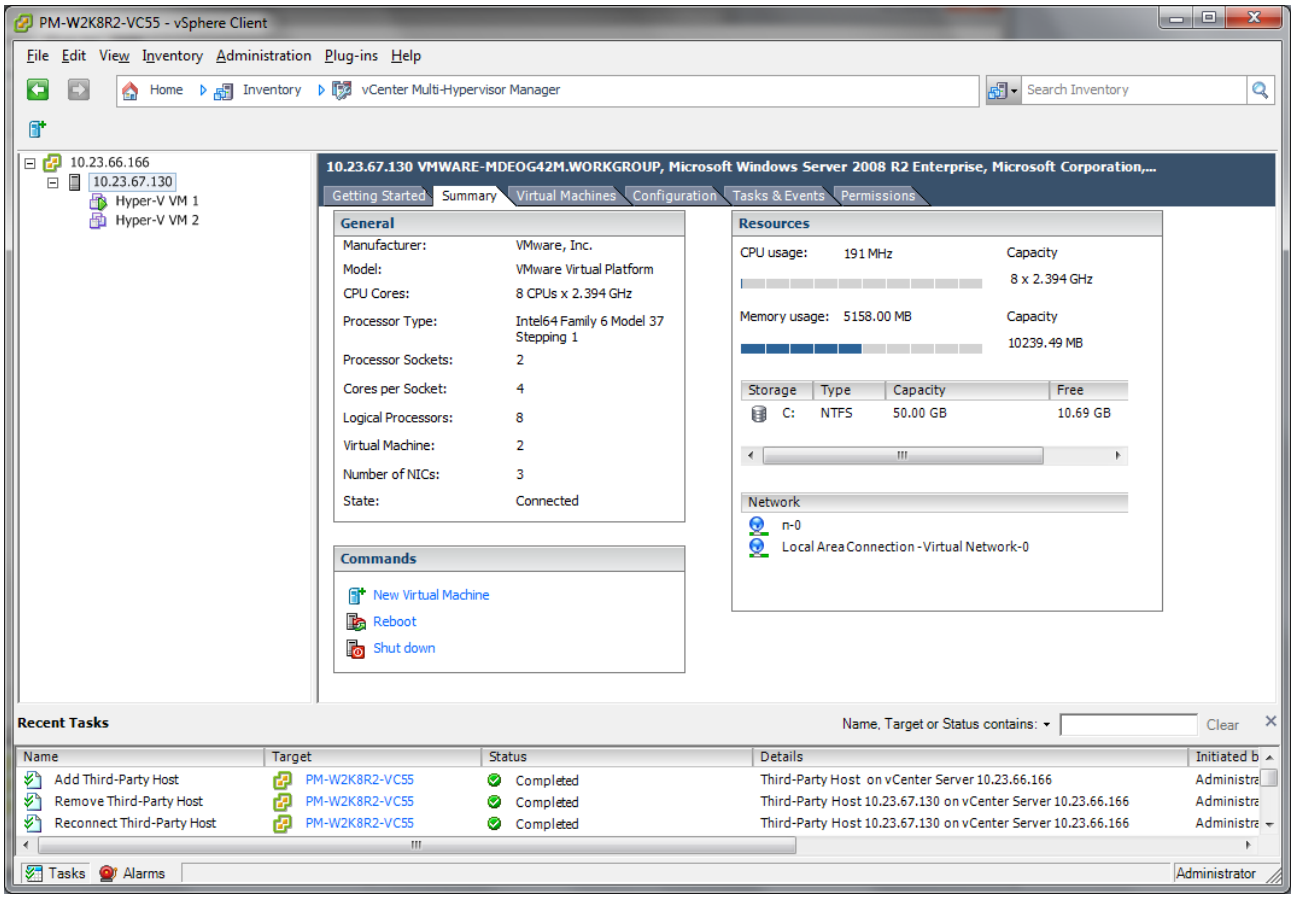


Рисунок 3.2– Користуватське вікно VMware vSphere [13]

Вузли vSphere ESXi можна підключити до домену Active Directory. Після цього Active Directory зможе виконувати автентифікацію користувачів і усуне

необхідність у створенні локальних облікових записів користувача на кожному вузлі.

Поєднання можливостей профілів вузлів, Image Builder і середовища PXE в одному рішенні VMware vSphere Auto Deploy спрощує встановлення і оновлення сайтів. В бібліотеці Auto Deploy централізовано зберігаються образи вузлів vSphere. Адміністратори мають можливість автоматично ініціалізувати нові вузли на базі визначених користувачем правил.

Обмеження інфраструктури:

- Максимум RAM гостьової системи: 4 ТБ;
- Максимум RAM хоста: 6 ТБ;
- Максимальна кількість хостів в кластері: 64;
- Максимальна кількість процесорів на віртуальну машину: 128;
- Максимальна кількість процесорів на хост: 480;
- Максимальна кількість віртуальних CPU на один фізичний CPU: 32;
- Максимальна кількість віртуальних машин на один хост: 1024;
- Максимальна кількість віртуальних CPU на одну віртуальну машину: 4;
- Максимальна кількість RAM на гостьову віртуальну машину: 64 ГБ;
- Максимальний розмір тома: 64 ТБ.

ESX Server має компонент брандмауера без збереження стану, який налаштовується за допомогою клієнта vSphere або через командний рядок ESXCLI. Модуль брандмауера використовує набори правил портів, що визначаються адміністраторами для кожної служби. Додатково можна задати діапазони або окремі IP-адреси, яким дозволений доступ до служб вузла.

3.3 XEN

Для управління гіпервізором XEN існує декілька рішень:

- libvirt;
- oVirt;

- Enomalism;
- OpenECP;
- ConVirt;
- OpenNEbula;

Велика кількість рішень продиктована тим що XEN це безплатний гіпервізор. Також кожне рішення має свій набір функціоналу. Зупинимося на рішенні з найбільш привабливим функціоналом.

ConVirt Open Source пропонує такі можливості для управління гіпервізором XEN:

- високопродуктивний веб інтерфейс який оновлюється в режимі реального часу;
- централізований репозиторій для зберігання даних;
- не потребує встановлення агентів на фізичні сервери з віртуальними машинами;
- має підтримку розділення прав доступу між адміністраторами системи;
- підтримує моніторинг віртуальних машин на фізичних серверах;
- підтримує використання зовнішніх СЗД;
- має підтримку міграції ВМ між фізичними серверами.

Вікно користувацького інтерфейсу ConVirt зображено на рисунку 3.3.

Рисунок 3.3 – Вікно користувацького інтерфейсу ConVirt [14]

3.4 KVM

Для гіпервізора KVM існує система управління Virtual Machine Manager яка була розроблена компанією Red Hat.

Її можливості наступні:

– створення віртуальної машини;
– запуск віртуальної машини;
– зупинка віртуальної машини;
– збір інформації про продуктивність ресурсів на конкретній віртуальній машині;
– збір інформації про продуктивність ресурсів на конкретному фізичному сервері.

4 РОЗРОБКА СИСТЕМИ УПРАВЛІННЯ ВМ В ІТ-ІНФРАСТРУКТУРІ

4.1 Загальний опис

Основною ціллю системи управління є міграція віртуальних машин. Також система може пропонує можливості по моніторингу серверів. Інформація по моніторингу дозволить адміністратору системи контролювати стан фізичних серверів. У адміністратора також є можливість виконувати прості команди. Завдяки цьому СУ значно розширює свої функціональні можливості.

Система управління міграцією віртуальних машин включає в себе сім модулів. Ця система управління зображена на структурній схемі рисунку 4.1 та додатку А.

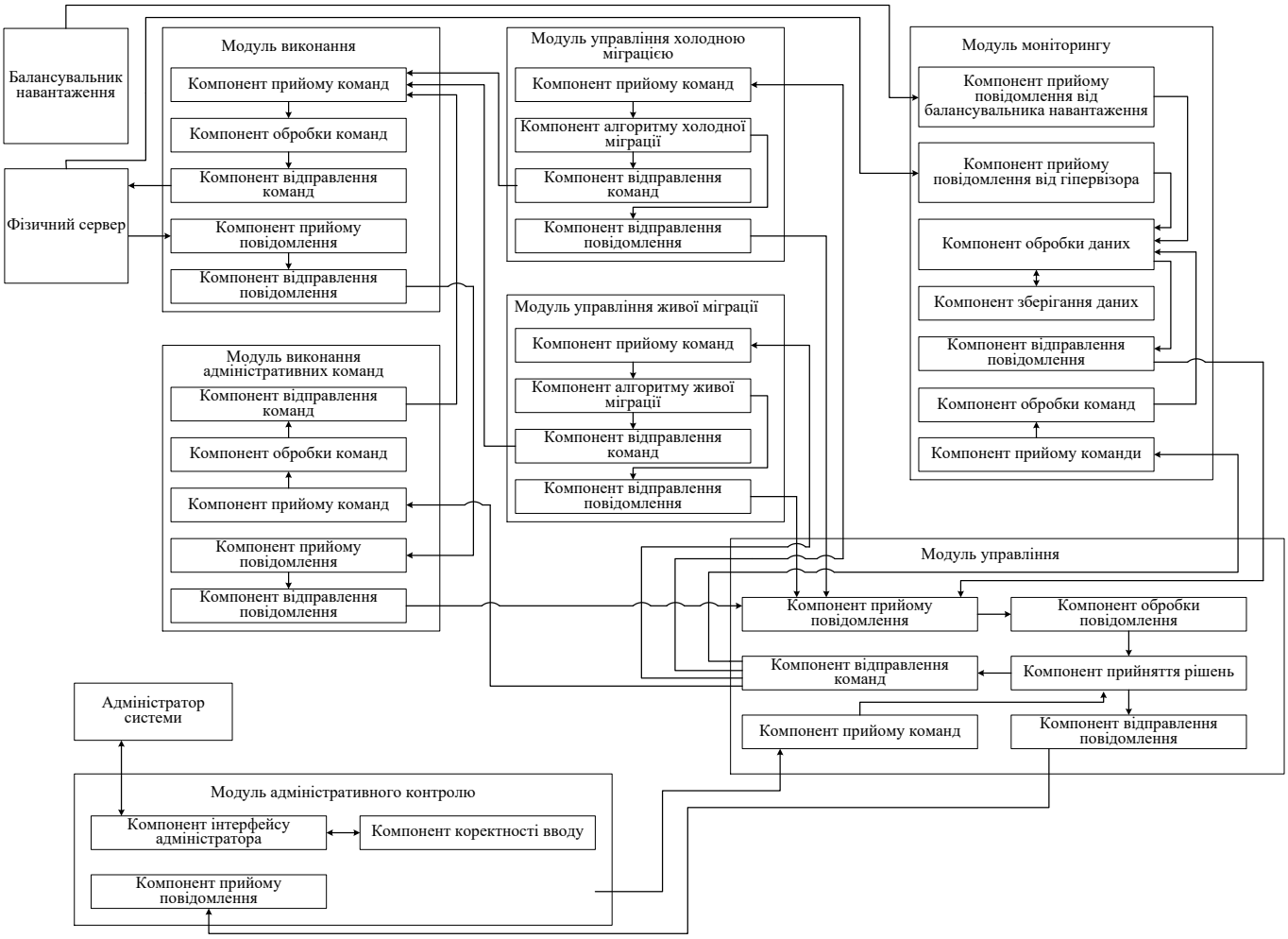


Рисунок 4.1 – Система управління віртуальних машин

Модулі які входять до складу системи управління:

- модуль моніторингу;
- підсистема управління;
- модуль адміністративного контрою;
- модуль управління холодної міграції;
- модуль управління гарячої міграції;
- модуль виконання адміністративних команд;
- модуль виконання.

4.2 Модуль моніторингу.

Завдання модулю моніторингу це отримання інформації від фізичних серверів, балансувальника навантаження та гіпервізорів з віртуальними машинами.

Модуль моніторингу складеться з наступних компонентів:

– компонент прийому повідомлення від балансувальника навантаження;
– компонент прийому повідомлення від гіпервізора;
– компонент обробки даних;
– компонент зберігання даних;
– компонент відправлення повідомлення;
– компонент обробки команд;
– компонент прийому команди.

До модуля моніторингу надходить наступна інформація від фізичного серверу:

– кількість навантаження у CPU та його ядер;
– об’єм зайнятої RAM;
– об’єм вільної RAM;
– кількість навантаження у мережевих адаптерів;
– робочий стан фізичного серверу.

До модуля моніторингу надходить наступна інформація від гіпервізора:

– кількість задіяних віртуальних машин;
– кількість RAM яка виділена під кожну віртуальну машину;
– об’єм RAM яка виділяться під віртуальну машину;
– кількість ядер CPU які виділені під кожну віртуальну машину;
– кількість зайнятих ядер CPU кожної віртуальної машини;
– фізичні пристрої до яких звертається кожна віртуальна машина;
– робочий стан гіпервізору.

До модуля моніторингу надходить наступна інформація від віртуальної машини:

– відсоток завантаженості ядер CPU які виділені під віртуальну машину;
– кількість зайнятого простору у системі зберігання даних;
– робочий стан процесів які виконуються у ВМ;
– робочий стан ВМ.

Компонент прийому повідомлення від балансувальника навантаження отримуючи дані направляє їх до компоненту обробки даних. Це саме стосується і компоненту прийому повідомлення від гіпервізора. Компонент обробки даних перетворює дані отримані від цих двух компонентів у більш зручну для зберігання форму та направляє ці дані до компоненту зберігання даних. Якщо потрібно дані моніторингу передати далі, то отримуючи необхідну команду компонент прийому команди направляє команду до компоненту обробки команд. Вже оброблена команда у вигляді, яка зрозуміла компоненту обробки даних виконуються цим компонентом і дані попередньо отримані дані від компонента зберігання дані надходять до компонента відправлення повідомлення. І потім компонент відправлення повідомлення надсилає дані далі до модуля управління.

4.3 Модуль адміністративного контролю

Модуль адміністративного контролю являє собою інтерфейс між бізнес-логікою системи та адміністратором системи.

Модуль адміністративного контролю складається з наступних компонентів:

- Компонент інтерфейсу користувача;
- Компонент коректності вводу;
- Компонент прийому повідомлення;
- Компонент відправлення команд.

До модуля адміністративного контролю надходить наступна інформація від підсистеми управління:

- данні які надходять з модуля моніторингу;
- результати роботи модуля холодної міграції;
- результати роботи модуля гарячої міграції;
- результати виконання модуля адміністративних команд;
- статус виконання модуля холодної міграції;
- статус виконання модуля гарячої міграції;
- статус виконання модуля адміністративних команд.

До модуля адміністративного контролю надходить наступна інформація від адміністратора системи:

– команда на виконання холодної міграції;
– команда на виконання гарячої міграції;
– команда на виконання адміністративних команд.

Компонент інтерфейсу користувача отримуючи команди від адміністратора системи відправляє їх до компоненту коректності вводу. Там ці дані проходять перевірку. Якщо данні які буди надіслані адміністратором системи коректні, то ці дані відправляються до компоненту відправлення команд. Якщо дані були некоректними, то компонент інтерфейсу користувача генерує повідомлення про помилку і надсилає його до компоненту інтерфейсу адміністратора. Коректні команди пересилаються до модулю управління за допомогою компонента відправлення команд.

4.4 Модуль управління холодної міграції

Компонент прийому команд модуля управління холодної міграції отримає команди від модуля управління. Потім ці команди надсилаються до компоненту алгоритму холодної міграції. Алгоритм компоненту холодної міграції зображено на рисунку 4.2 та додатку Б.

Компонент працює за наступними етапами алгоритму:

Етап 1. Отримання команди від компоненту прийому команд. Команда обробляється на коректність. Якщо команда не коректна, алгоритм сповіщає адміністратора системи за допомогою компонента відправлення повідомлення.

Етап 2. Якщо надійшла команда з параметрами вибрати сервер автоматично, тоді алгоритм перевіряє наявність вільних ресурсів пулу серверів. Якщо вільні ресурси не знайдено, алгоритм сповіщає адміністратора, а якщо вільні ресурси наявні, то алгоритм вибирає найменш навантажений сервер. Якщо надійшла команда з параметром ручного вибору фізичного серверу, алгоритм перевіряє доступність серверу за IP-адресую. Якщо сервер за такою адресую не знайдено, алгоритм

направляє відповідне повідомлення адміністратору системи, в іншому випадку починається перевірка наявності вільних ресурсів вказаного адміністратором сервера. Якщо вільних ресурсів для процесу міграції не достатньо, алгоритм сповіщає про це адміністратора. Якщо ресурси наявні, то починається наступний етап роботи алгоритму.

Етап 3. Кількість ресурсів які необхідні віртуальній машині резервуються на цільовому фізичному сервері.

Етап 4. Відбувається зупинка ВМ, яка була створена на початковому фізичному сервері.

Етап 5. Відбувається створення ВМ на цільовому фізичному сервері.

Етап 6. Перенос вмісту віртуального диску ВМ на цільовий сервер.

Етап 7. Старт нової ВМ на цільовому сервері згідно з параметрами конфігураційного файла віртуальної машини, а також використовуючи вміст віртуального диска.

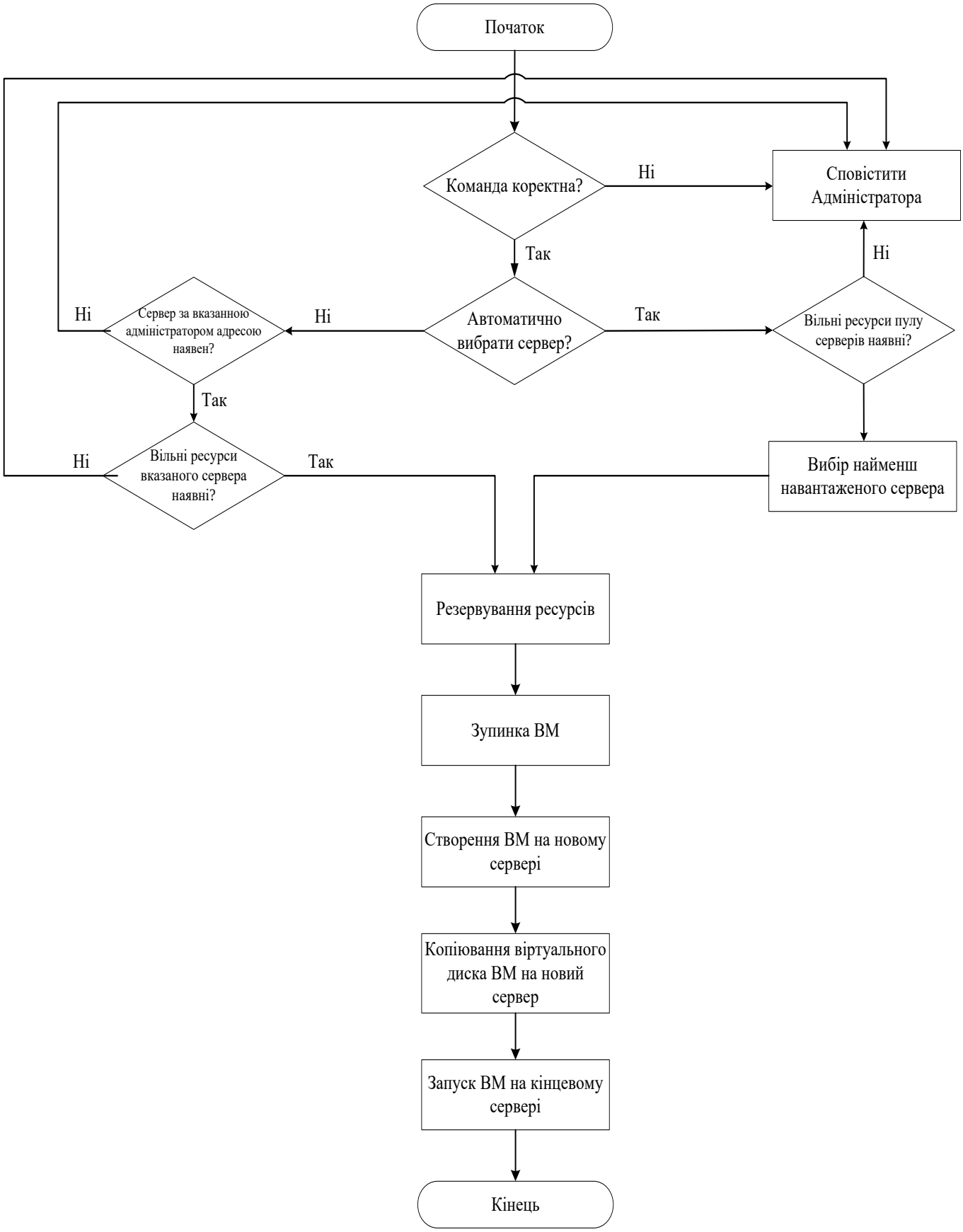


Рисунок 4.2 – Блок схема модуля управління холодної міграції

4.5 Модуль управління живої міграції

Компонент прийому команд модуля управління живої міграції отримає команди від модуля управління. Потім ці команди надсилаються до компоненту алгоритму живої міграції. Алгоритм компоненту холодної міграції зображено на рисунку 4.3 та додатку В.

Компонент працює за наступними етапами алгоритму:

Етап 1. Отримання команди від компоненту прийому команд. Команда обробляється на коректність. Якщо команда не коректна, алгоритм сповіщає адміністратора системи за допомогою компонента відправлення повідомлення.

Етап 2. Якщо надійшла команда з параметрами вибрати сервер автоматично, тоді алгоритм перевіряє наявність вільних ресурсів пулу серверів. Якщо вільні ресурси не знайдено, алгоритм сповіщає адміністратора, а якщо вільні ресурси наявні, то алгоритм вибирає найменш навантажений сервер. Якщо надійшла команда з параметром ручного вибору фізичного серверу, алгоритм перевіряє доступність серверу за IP-адресую. Якщо сервер за такою адресую не знайдено, алгоритм направляє відповідне повідомлення адміністратору системи, в іншому випадку починається перевірка наявності вільних ресурсів вказаного адміністратором сервера. Якщо вільних ресурсів для процесу міграції не достатньо, алгоритм сповіщає про це адміністратора. Якщо ресурси наявні, то починається наступний етап роботи алгоритму.

Етап 3. Кількість ресурсів які необхідні віртуальній машині резервуються на цільовому фізичному сервері.

Етап 4. Відбувається створення ВМ на цільовому фізичному сервері.

Етап 5. Перенос вмісту віртуального диску ВМ на цільовий сервер.

Етап 6. Дані з RAM переносяться на цільовий сервер.

Етап 7. Відбувається зупинка ВМ, яка була створена на початковому фізичному сервері.

Етап 8. Старт нової ВМ на цільовому сервері згідно з параметрами конфігураційного файла віртуальної машини.

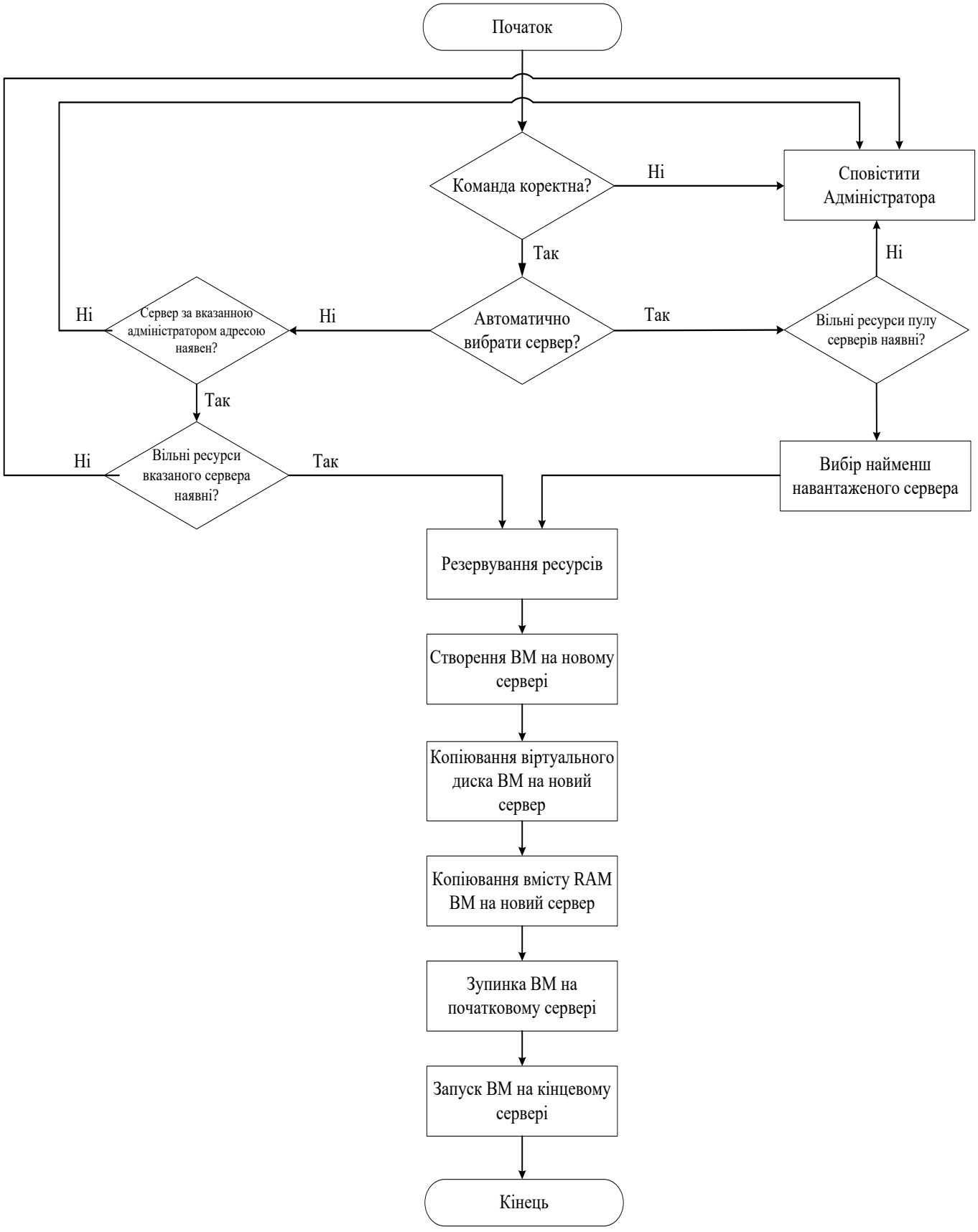


Рисунок 4.3 – Блок схема модуля управління живої міграції

4.6 Модуль виконання адміністративних команд

Модуль виконання адміністративних команд складається з наступних компонентів:

- Компонент відправлення команд;
- Компонент обробки команд;
- Компонент прийому команд;
- Компонент прийому повідомлення;
- Компонент відправлення повідомлення.

Команди які підтримує модуль виконання адміністративних команд:

- зупинка фізичного серверу;
- статус роботи гіпервізора;
- запуск фізичного сервера;
- перезавантаження фізичного сервера;
- зміна мережевого порту;
- статус роботи компонентів апаратного обладнання фізичного серверу.

Команди які надходять до модуля виконання адміністративних команд з модуля управління приймаються компонентом прийому команд. Перед тим як відправити команді до модулю виконання компонентом відправлення команд, команди мають бути оброблені компонентом обробки команд.

4.7 Модуль виконання

Модуль виконання складається з складається з наступних компонентів:

- Компонент відправлення команд;
- Компонент обробки команд;
- Компонент прийому команд;
- Компонент прийому повідомлення;
- Компонент відправлення повідомлення.

Компонент прийому команд модуля виконання приймає команди від модуля управління холодною міграцією, модуля управління живої міграції, модуля виконання адміністративних команд. Далі команди обробляються компонентом обробки команд за для того, щоб бути зрозумілими фізичному серверу. За відправлення оброблених команд відповідає компонент відправлення команд. А за прийом даних від фізичного серверу відповідає компонент прийому повідомлення.

4.8 Модуль управління

Модуль управління складається з шести компонентів:

- Компонент прийому повідомлення;
- Компонент відправлення команд;
- Компонент прийому команд;
- Компонент обробки повідомлення;
- Компонент прийняття рішень;
- Компонент відправлення повідомлення.

Компонент прийому повідомлення приймає повідомлення з модуля управління холодної міграції, модуля управління живої міграції, модуля виконання адміністративних команд, модуля моніторингу. Потім ці повідомлення надсилаються в компонент обробки повідомлення і вже в обробленому вигляді надсилаються да компоненту прийняття рішень. Компонент відправлення команд надсилає команди до модуля управління холодної міграції, модуля управління живої міграції, модуля виконання адміністративних команд, модуля моніторингу. Компонент прийняття рішень крім того що приймає команди від модуля адміністративного контролю, та надсилає туди повідомлення, цей компонент аналізує обчислювальну активність на фізичних серверах пулу серверів. В результаті аналізу компонент прийняття рішень може самостійно обирати варіанти між алгоритмами міграції та переміщувати віртуальні машини між фізичними серверами.

4.9 Сценарії використання системи

Система управління дозволяє автоматично застосовувати алгоритми міграції в залежності від навантаження. Але можуть виникати ситуації коли втручання системного адміністратора необхідне. Наприклад, для встановлення системи, початкового налаштування, зміни налаштувань при необхідності, обробка помилок в роботі системи. Також, ручне управління системою дає змогу більш ефективно розподіляти навантаження між фізичними серверами в ситуаціях, коли СУ не має достатньої кількості даних. На діаграмі Use case зображено всі можливості які має адміністратор при роботі з СУ.

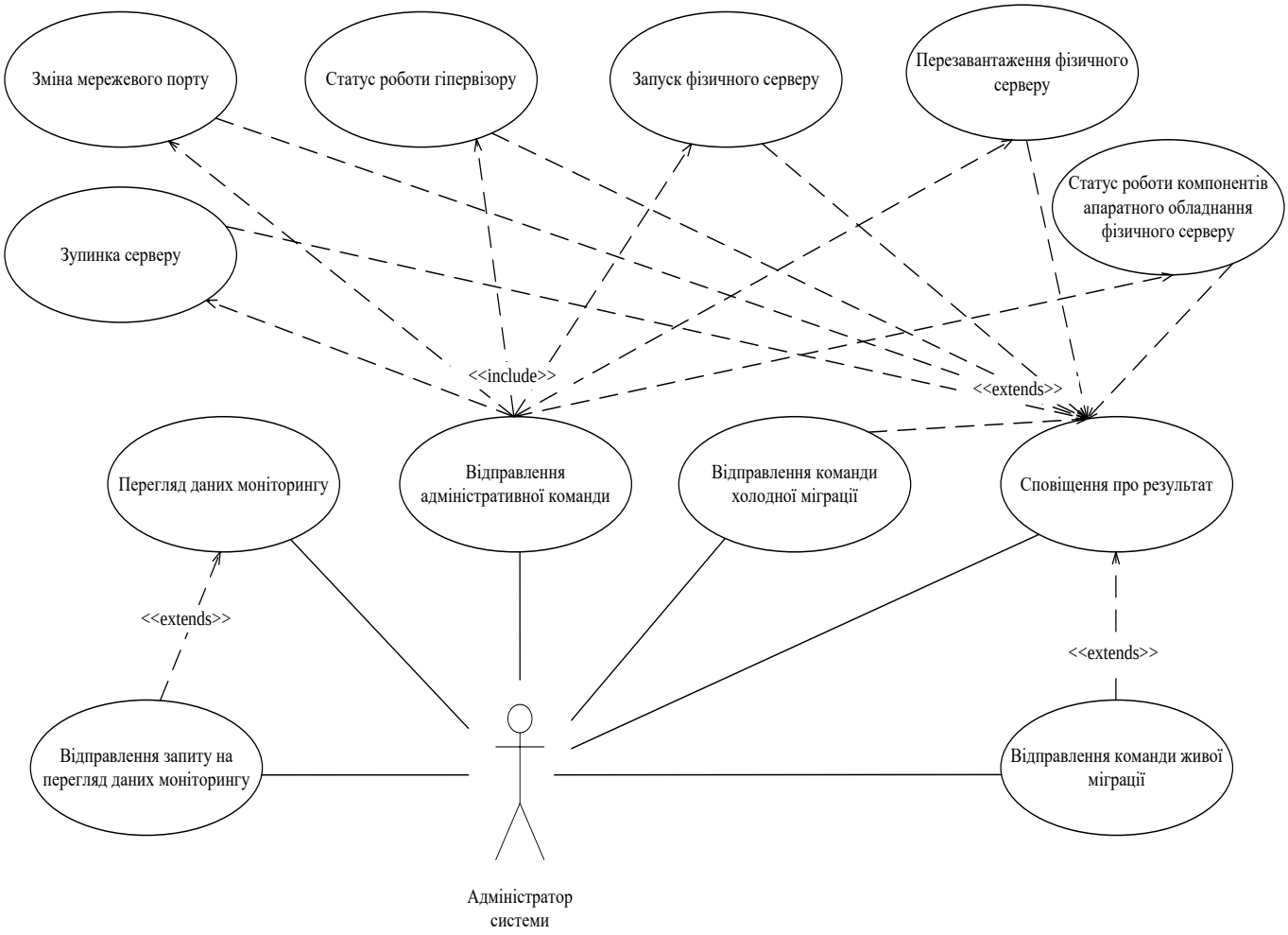


Рисунок 4.4 – Use case діаграма

Ім’я	Відправлення запиту на перегляд даних моніторингу.
Номер	UC_001
Опис	Адміністратор системи хоче переглянути дані моніторингу і отримати їх після вводу потрібної команди.
Актори	Адміністратор системи.
Переваги	Це дозволить більш ретельно слідкувати за станом фізичних серверів, рівнем навантаження кожного з серверу та загальним рівнем навантаження у пулі серверів. Маючи таку інформацію можна додавати фізичні сервери до загального пулу серверів, або відключати фізичні сервери, які простоюють за для економії.
Частота використання	6 разів на день.
Дії актора	Адміністратор вводить команду в терміналі з бажаними параметрами.
Початковий стан	Термінал повинен бути готовий до прийому команди.
Кінцевий стан	Вивід на екран терміналу даних моніторингу згідно з параметрами команди.
Головний шлях	1.Компонент коректності вводу перевіряє команду та параметри команди. 2. Компонент відправлення команди передає команду до модуля управління. 3. Компонент прийняття рішень перевіряє тип команди та використовуючи власний компонент відправлення команд, відправляє команду та параметри до модулю моніторингу. 4. Модуль моніторингу прийнявши команду та параметри, обробляє їх використовуючи власний компонент обробки команд, та надсилає команди до компоненту обробки даних. 5. Компонент обробки даних отримує необхідні данні від

	компоненту зберігання даних, обробляє їх і надсилає до модуля моніторингу через власний компонент відправлення команд. 6. Модуль управління прийнявши повідомлення, обробляє його та надсилає до модуля адміністративного контролю через власний компонент відправлення повідомлення. 7. Компонент прийому повідомлення модуля адміністративного контролю надсилає повідомлення до компонента інтерфейсу адміністратора. 8. Компонент інтерфейсу адміністратора демонструє повідомлення адміністратору системи.
Альтернативні шляхи	
Виключення	Необхідні запитувані дані відсутні 1. Система повідомляє адміністратора про відсутність запитуваних даних.

Ім’я	Відправлення адміністративної команди – зупинка фізичного серверу.
Номер	UC_002
Опис	Адміністратор системи хоче зупинити конкретний фізичний сервер.
Актори	Адміністратор системи.
Переваги	Це дозволить зупиняти сервери обчислювальні ресурси яких простоюють.
Частота використання	2 рази на день.
Дії актора	Адміністратор вводить команду в термінал, вказуючи IP-адресу

	цільового фізичного серверу.
Початковий стан	Термінал повинен бути готовий до прийому команди. Цільовий фізичний сервер має бути увімкнений.
Кінцевий стан	Цільовий фізичний сервер має бути вимкнений.
Головний шлях	1.Компонент коректності вводу перевіряє команду та параметри команди. 2. Компонент відправлення команди передає команду до модуля управління. 3. Компонент прийняття рішень перевіряє тип команди та використовуючи власний компонент відправлення команд, відправляє команду та параметри до модуля виконання адміністративних команд. 4. Модуль виконання адміністративних команд прийнявши команду та параметри, обробляє їх використовуючи власний компонент обробки команд, та надсилає команди до модулю виконання. 5. Модуль виконання прийнявши команди, обробляє їх використовуючи власний компонент обробки команд, та надсилає команди до кінцевого фізичного серверу.
Альтернативні шляхи	
Виключення	Фізичного серверу по вказаній IP-адресі не існує. 1. Система повідомляє адміністратора про відсутність фізичного серверу по вказаній IP-адресі. Фізичний сервер вже вимкнутий. 1. Система повідомляє адміністратора про те, що сервер вже вимкнутий.

Ім’я	Відправлення адміністративної команди – зміна мережевого порту.
Номер	UC_003
Опис	Адміністратор системи хоче змінити мережевий порт СУ.
Актори	Адміністратор системи.
Переваги	Це дозволить уникати мережевих конфліктів з іншим програмним забезпеченням.
Частота використання	Дуже рідко.
Дії актора	Адміністратор вводить команду в термінал, вказуючи новий мережевий порт.
Початковий стан	Термінал повинен бути готовий до прийому команди. Новий мережевий порт моє бути вільним.
Кінцевий стан	Мережевий порт має бути змінений на новий.
Головний шлях	1.Компонент коректності вводу перевіряє команду та параметри команди. 2. Компонент відправлення команди передає команду до модуля управління. 3. Модуль управління додає команду до черги виконання і припиняє додавати нові команди. 4. Після виконання всіх команд у черзі модуль управління перезавантажує СУ. 5. Модуль управління через власний компонент відправлення повідомлення надсилає до модуля адміністративного контролю повідомлення про успішність виконання команди. 6. Компонент прийому повідомлення модуля адміністративного контролю надсилає повідомлення до компонента інтерфейсу адміністратора. 7. Компонент інтерфейсу адміністратора демонструє повідомлення

	адміністратору системи.
Альтернативні шляхи	
Виключення	Новий мережевий порт співпадає з старим. 1. Система повідомляє адміністратора про співпадання нового мережевого порту з старим. 2. Компонент інтерфейсу користувача просить адміністратора вказати інший порт. Новий мережевий порт вже зайнятий. 1. Система повідомляє адміністратора про те, що новий мережевий порт вже зайнятий. 2. Компонент інтерфейсу користувача просить адміністратора вказати інший порт.

Ім’я	Відправлення адміністративної команди – запуск фізичного серверу.
Номер	UC_004
Опис	Адміністратор системи хоче запустити конкретний фізичний сервер
Актори	Адміністратор системи.
Переваги	Це дозволить запускати фізичні сервери коли обчислювальних ресурсів пулу серверів недостатньо.
Частота використання	2 рази на день.
Дії актора	Адміністратор вводить команду в термінал, вказуючи IP-адресу цільового фізичного серверу.

Початковий стан	Термінал повинен бути готовий до прийому команди. Цільовий фізичний сервер має бути вимкнений.
Кінцевий стан	Цільовий фізичний сервер має бути увімкнений.
Головний шлях	1.Компонент коректності вводу перевіряє команду та параметри команди. 2. Компонент відправлення команди передає команду до модуля управління. 3. Компонент прийняття рішень перевіряє тип команди та використовуючи власний компонент відправлення команд, відправляє команду та параметри до модуля виконання адміністративних команд. 4. Модуль виконання адміністративних команд прийнявши команду та параметри, обробляє їх використовуючи власний компонент обробки команд, та надсилає команди до модулю виконання. 5. Модуль виконання прийнявши команди, обробляє їх використовуючи власний компонент обробки команд, та надсилає команди до кінцевого фізичного серверу.
Альтернативні шляхи	
Виключення	Фізичного серверу по вказаній IP-адресі не існує. 1. Система повідомляє адміністратора про відсутність фізичного серверу по вказаній IP-адресі. Фізичний сервер вже запущений. 1. Система повідомляє адміністратора про те, що сервер вже запущений.

Ім’я	Відправлення адміністративної команди – перезавантаження фізичного серверу.
Номер	UC_005
Опис	Адміністратор системи хоче перезавантажити конкретний фізичний сервер
Актори	Адміністратор системи.
Переваги	Це дозволить перезавантажувати фізичні сервери коли вони стають недієздатними в разі програмний помилок.
Частота використання	1 раз в місяць.
Дії актора	Адміністратор вводить команду в термінал, вказуючи IP-адресу цільового фізичного серверу.
Початковий стан	Термінал повинен бути готовий до прийому команди. Цільовий фізичний сервер має бути увімкнений.
Кінцевий стан	Цільовий фізичний сервер має бути увімкнений.
Головний шлях	1.Компонент коректності вводу перевіряє команду та параметри команди. 2. Компонент відправлення команди передає команду до модуля управління. 3. Компонент прийняття рішень перевіряє тип команди та використовуючи власний компонент відправлення команд, відправляє команду та параметри до модуля виконання адміністративних команд. 4. Модуль виконання адміністративних команд прийнявши команду та параметри, обробляє їх використовуючи власний компонент обробки команд, та надсилає команди до модулю виконання. 5. Модуль виконання прийнявши команди, обробляє їх використовуючи власний компонент обробки команд, та надсилає

	команди до кінцевого фізичного серверу.
Альтернативні шляхи	
Виключення	Фізичного серверу по вказаній IP-адресі не існує. 1. Система повідомляє адміністратора про відсутність фізичного серверу по вказаній IP-адресі. Фізичний сервер вимкнутий. 1. Система повідомляє адміністратора про те, що сервер вимкнутий.

Ім’я	Відправлення запиту на статус роботи компонентів апаратного обладнання.
Номер	UC_006
Опис	Адміністратор системи хоче переглянути статус роботи компонентів апаратного обладнання і отримати його після вводу потрібної команди.
Актори	Адміністратор системи.
Переваги	Це дозволить більш ретельно слідкувати за станом фізичних серверів, та за рівнем працьоспроможності кожного з серверу та його компонентами у пулі серверів. Маючи таку інформацію можна проводити заміну непрацюючого обладнання.
Частота використання	6 разів на день.
Дії актора	Адміністратор вводить команду в терміналі з бажаними параметрами.
Початковий стан	Термінал повинен бути готовий до прийому команди.
Кінцевий стан	Вивід на екран терміналу даних статусу роботи згідно з параметрами команди.

Головний шлях	1.Компонент коректності вводу перевіряє команду та параметри команди. 2. Компонент відправлення команди передає команду до модуля управління. 3. Компонент прийняття рішень перевіряє тип команди та використовуючи власний компонент відправлення команд, відправляє команду та параметри до модулю моніторингу. 4. Модуль моніторингу прийнявши команду та параметри, обробляє їх використовуючи власний компонент обробки команд, та надсилає команди до компоненту обробки даних. 5. Компонент обробки даних отримує необхідні данні від компоненту зберігання даних, обробляє їх і надсилає до модуля моніторингу через власний компонент відправлення команд. 6. Модуль управління прийнявши повідомлення, обробляє його та надсилає до модуля адміністративного контролю через власний компонент відправлення повідомлення. 7. Компонент прийому повідомлення модуля адміністративного контролю надсилає повідомлення до компонента інтерфейсу адміністратора. 8. Компонент інтерфейсу адміністратора демонструє повідомлення адміністратору системи.
Альтернативні шляхи	
Виключення	Необхідні запитувані дані відсутні 1. Система повідомляє адміністратора про відсутність запитуваних даних.

Ім’я	Відправлення команди холодної міграції.
Номер	UC_007
Опис	Адміністратор системи хоче в ручному режимі перемістити віртуальну машину з одного фізичного серверу на інший фізичний сервер за використовуючи холодну міграцію.
Актори	Адміністратор системи.
Переваги	Це дозволить більш гнучко розподіляти ресурси, керуючись даними яких немає СУ.
Частота використання	1 раз на день.
Дії актора	Адміністратор вводить команду в терміналі з бажаними параметрами.
Початковий стан	Віртуальна машина має існувати на початковому сервері.
Кінцевий стан	Віртуальна машина має існувати на кінцевому сервері.
Головний шлях	1.Компонент коректності вводу перевіряє команду та параметри команди. 2. Компонент відправлення команди передає команду до модуля управління. 3. Компонент прийняття рішень перевіряє тип команди та використовуючи власний компонент відправлення команд, відправляє команду та параметри до модулю управління холодною міграцією. 4. Модуль управління холодною міграцією прийнявши команду та параметри, обробляє їх використовуючи власний компонент обробки команд, та надсилає команди до компоненту алгоритму холодної міграції. 5. Компонент алгоритму холодної міграції передає команди по черзі до компоненту відправлення команд.

	6. Компонент відправлення команд надсилає команди до модуля виконання. 7. Модуль виконання обробляє команди і надсилає команди до фізичного серверу.
Альтернативні шляхи	
Виключення	Фізичного сервер відсутній. 1. Система повідомляє адміністратора про відсутність фізичного серверу. Віртуальна машина не знайдена. 1. Система повідомляє адміністратора про відсутність віртуальної машини.

Ім’я	Відправлення команди живої міграції.
Номер	UC_008
Опис	Адміністратор системи хоче в ручному режимі перемістити віртуальну машину з одного фізичного серверу на інший фізичний сервер за використовуючи живу міграцію.
Актори	Адміністратор системи.
Переваги	Це дозволить більш гнучко розподіляти ресурси, керуючись даними яких немає СУ.
Частота використання	5 разів на день.
Дії актора	Адміністратор вводить команду в терміналі з бажаними параметрами.

Початковий стан	Віртуальна машина має існувати на початковому сервері.
Кінцевий стан	Віртуальна машина має існувати на кінцевому сервері.
Головний шлях	1.Компонент коректності вводу перевіряє команду та параметри команди. 2. Компонент відправлення команди передає команду до модуля управління. 3. Компонент прийняття рішень перевіряє тип команди та використовуючи власний компонент відправлення команд, відправляє команду та параметри до модулю управління живої міграції. 4. Модуль управління живої міграції прийнявши команду та параметри, обробляє їх використовуючи власний компонент обробки команд, та надсилає команди до компоненту алгоритму холодної міграції. 5. Компонент алгоритму живої міграції передає команди по черзі до компоненту відправлення команд. 6. Компонент відправлення команд надсилає команди до модуля виконання. 7. Модуль виконання обробляє команди і надсилає команди до фізичного серверу.
Альтернативні шляхи	
Виключення	Фізичного сервер відсутній. 1. Система повідомляє адміністратора про відсутність фізичного серверу. Віртуальна машина не знайдена. 1. Система повідомляє адміністратора про відсутність віртуальної машини.

Ім’я	Відправлення запиту на статус роботи гіпервізору
Номер	UC_009
Опис	Адміністратор системи хоче переглянути статус роботи гіпервізору і отримати його після вводу потрібної команди.
Актори	Адміністратор системи.
Переваги	Це дозволить більш ретельно слідкувати за віртуальних машин, та за рівнем працьоспроможності кожної з віртуальних машин у гіпервізорі. Маючи таку інформацію можна більш гнучко застосовувати алгоритми міграції.
Частота використання	10 разів на день.
Дії актора	Адміністратор вводить команду в терміналі з бажаними параметрами.
Початковий стан	Термінал повинен бути готовий до прийому команди.
Кінцевий стан	Вивід на екран терміналу даних статусу роботи згідно з параметрами команди.
Головний шлях	1.Компонент коректності вводу перевіряє команду та параметри команди. 2. Компонент відправлення команди передає команду до модуля управління. 3. Компонент прийняття рішень перевіряє тип команди та використовуючи власний компонент відправлення команд, відправляє команду та параметри до модулю моніторингу. 4. Модуль моніторингу прийнявши команду та параметри, обробляє їх використовуючи власний компонент обробки команд, та надсилає команди до компоненту обробки даних.

	5. Компонент обробки даних отримує необхідні данні від компоненту зберігання даних, обробляє їх і надсилає до модуля моніторингу через власний компонент відправлення команд. 6. Модуль управління прийнявши повідомлення, обробляє його та надсилає до модуля адміністративного контролю через власний компонент відправлення повідомлення. 7. Компонент прийому повідомлення модуля адміністративного контролю надсилає повідомлення до компонента інтерфейсу адміністратора. 8. Компонент інтерфейсу адміністратора демонструє повідомлення адміністратору системи.
Альтернативні шляхи	
Виключення	Необхідні запитувані дані відсутні 1. Система повідомляє адміністратора про відсутність запитуваних даних.

Ім’я	Сповіщення про оезультат
Номер	UC_010
Опис	Адміністратор системи хоче отримати від системи повідомлення після виконання команд.
Актори	Адміністратор системи.
Переваги	Це дозволить адміністратору системи бути впевненим, що його команди виконуються.
Частота використання	25 разів на день.

Дії актора	Адміністратор споглядає на термінал
Початковий стан	Отримання результатів.
Кінцевий стан	Вивід на екран терміналу повідомлення.
Головний шлях	1. Модуль управління надсилає повідомлення до модуля адміністративного контролю використовуючи власний компонент відправлення повідомлення. 2. Компонент прийому повідомлення модуля адміністративного контролю надсилає повідомлення до компонента інтерфейсу адміністратора. 3. Компонент інтерфейсу адміністратора демонструє повідомлення адміністратору системи.
Альтернативні шляхи	Відправлення адміністративної команди. 1. Модуль виконання отримує повідомлення від фізичного серверу та надсилає його до модуля виконання адміністративних команд за допомогою власного компоненту відправлення повідомлень. 2. Модуль виконання адміністративних команд перенаправляє повідомлення до модуля управління за допомогою власного компоненту відправлення повідомлень. 3. Модуль управління прийнявши повідомлення, обробляє його. 4. Оброблене повідомлення через компонент прийняття рішень надходить до компоненту відправлення повідомлення. 5. Виконується головних шлях. Відправлення команди холодної міграції. 1. Модуль управління холодною міграцією надсилає повідомлення до модулю управління використовуючи власний компонент відправлення повідомлення. 2. Модуль управління прийнявши повідомлення, обробляє його.

	3. Оброблене повідомлення через компонент прийняття рішень надходить до компоненту відправлення повідомлення. 4. Виконується головних шлях. Відправлення команди живої міграції 1. Модуль управління живої міграції надсилає повідомлення до модулю управління використовуючи власний компонент відправлення повідомлення. 2. Модуль управління прийнявши повідомлення, обробляє його. 3. Оброблене повідомлення через компонент прийняття рішень надходить до компоненту відправлення повідомлення. 4. Виконується головних шлях.
Виключення	Необхідні запитувані дані відсутні 1. Система повідомляє адміністратора про відсутність запитуваних даних.

5 РОЗРОБЛЕННЯ СТАРТАП-ПРОЕКТУ

5.1 Опис ідеї проекту

5.1.1 Зміст ідеї

Зміст ідеї що пропонується, можливі напрямки застосування та основні вигоди, що може отримати користувач товару, описано у таблиці 4.1.

Таблиця 4.1

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Система управління міграцією віртуальних машин в корпоративній IT-інфраструктурі	Корпоративне застосування	Зменшення кількості необхідного апаратного обладнання при збереженні обчислювального навантаження

5.1.2 Аналіз потенційних техніко-економічних переваг ідеї

Таблиця 4.2 Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п/п	Техніко-економічні характеристики ідеї	(потенційні) товари/концепції конкурентів				W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		Мій проект						
1.	Технологічна собівартість товару (грн)	8400000				-	-	+

2.	Кількість разів використання	Безліч				-	-	+

5.2 Технологічний аудит проекту

Таблиця 4.3 – Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
	Зменшення кількості необхідного апаратного обладнання при збереженні обчислювального навантаження, або збільшення кількості апаратного навантаження без збільшення кількості фізичного обладнання	Програмний метод	Гіпервізор, балансувальник та віртуальні машини	Доступна
Обрана технологія реалізації ідеї проекту: Програмний метод реалізації				

5.3 Аналіз ринкових можливостей запуску стартап-проекту

5.3.1 Аналіз попиту

Таблиця 4.4 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	3
2	Загальний обсяг продаж, грн/ум.од	1 000 720 000 000 грн
3	Динаміка ринку (якісна оцінка)	Зростає

4	Наявність обмежень для входу (вказати характер обмежень)	немає
5	Специфічні вимоги до стандартизації та сертифікації	немає
6	Середня норма рентабельності в галузі (або по ринку), %	80%

Таким чином, можемо зробити висновок, що ринок є привабливим для проекту.

5.3.2 Потенційна групи клієнтів

Таблиця 4.5 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Збільшення потреб у обчислювальних ресурсах за одиницю грошей	Корпоративний сегмент ринку	Алгоритми міграції	Швидкодія

5.3.3 Аналіз ринкового середовища

Таблиця 4.6 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Вартість технології	Малий обсяг продажу	Зниження ціни
2			

Таблиця 4.7 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Знизити ціну	Підвищити обсяги продажу	Оновлення та покращення характеристик самої технології для заохочення клієнтів

5.3.4 Аналіз пропозиції

Таблиця 4.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Тип конкуренції: Олігополія	Великий відсоток зайнятого ринку від загального обсягу ринку	Підвищення якості та зменшення ціни
2. За рівнем конкурентної боротьби: Інтернаціональний	Велика зацікавленість в технології у всьому світі	Продаж малому та середньому бізнесу для привертання уваги.
3. За галузевою ознакою: внутрішньогалузева	Продукт розрахований на вузькоспеціалізований сектор ринку	Підвищення якості та зменшення ціни
4. Конкуренція за видами товарів: Товарно-видова	Можливість застосування на різному апаратному обладнанні	Продаж технології за ціною нижчою ніж за кордоном
5. За характером конкурентних переваг: нецінова	Унікальність алгоритмів	Вдосконалення власних алгоритмів
6. За інтенсивністю: Марочна	Використання ТМ	ТМ дозволить бути поміченим на ринку

5.3.5 Аналіз умов конкуренції в галузі

Таблиця 4.9 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	Amazon, Microsoft, VMware	Google			
Висновки:	Висока інтенсивність	Можливий вхід на ринок Конкуренти лише закордонні	Без постачальників	Диктують лише у тому, аби покращити деякі характеристики	Технологія в декілька разів дорожча

5.3.6 Перелік факторів конкурентоспроможності

Таблиця 4.10 – Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
	Економічний фактор	Вартість технології
	Рівень технології виробництва	Високий рівень складності виконання та вартість робіт
	Попит	Збільшення потреб у обчислювальних ресурсах за одиницю грошей

5.3.7 Аналіз сильних та слабких сторін стартап-проекту

Таблиця 4.11 – Порівняльний аналіз сильних та слабких сторін проекту

№ п/	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з … (назва

п			підприємства)						
			–3	–2	–1	0	+1	+2	+3
1	Економічний фактор	17	+						
2	Рівень технології виробництва	20		+					
3	Попит	15	+						

5.3.8 Матриця аналізу сильних та слабких сторін, загроз та можливостей

Таблиця 4.12 – SWOT-аналіз стартап-проекту

Сильні сторони: Висока продуктивність Багаторазове використання Простота використання	Слабкі сторони: Відсутність ТМ та патенту на технологію Проведення тестування та досвід
Можливості: Збільшення кількості можливостей Розширення галузевого впливу	Загрози: Висока конкуренція

5.3.9 Альтернативи ринкової поведінки

Таблиця 4.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Зниження вартості	70%	5-6
2	Залучення сил	90%	2-3

З означених альтернатив обираю ту, яка дає можливість залучити більше спеціалістів для покращення та вдосконалення технології для отримання ресурсів за менший строк та більшою ймовірністю.

5.4 Розроблення ринкової стратегії проекту

5.4.1 Визначення стратегії охоплення ринку

Таблиця 4.14 – Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи	Інтенсивність конкуренції в сегменті	Простота входу у сегмент

			(сегменту)		
1	концентрований маркетинг	Середня	Високий	Висока	Висока
Які цільові групи обрано: концентрований маркетинг					

5.4.2 Базова стратегія розвитку

Таблиця 4.15 – Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку*
1	Вдосконалення та нижча ціна	Стратегія спеціалізації	Нижча вартість	Стратегія зайняття конкурентної ніші

5.4.3 Вибір стратегії конкурентної поведінки

Таблиця 4.16 – Визначення базової стратегії конкретної поведінки

№ п/п	Чи є проект «першопроходьцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки*
	Ні	Ні	Не буде	Стратегія зайняття конкурентної ніші

5.4.4 Стратегія позиціонування

Таблиця 4.17 – Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)

1	Технологія, яка не доступна у конкурентів	Стратегія зайняття конкурентної ніші	Використання багаторазове, за придбання технології один раз	Нижча ціна ніж у конкурентів Вдосконалення та підтримка Потрібність на ринку України за відсутності альтернатив

5.5 Розроблення маркетингової програми стартап-проекту

5.5.1 Формування маркетингової концепції проекту

Таблиця 4.18 – Визначення ключових переваг концепції потенційного проекту

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
1	Зменшення кількості необхідного апаратного обладнання при збереженні обчислювального навантаження	Вдосконалення, постійна підтримка технології, відсутність альтернатив та конкурентна ціна	Ціна Вдосконалення та підтримка вже після придбання

5.5.2 Трирівнева маркетингова модель товару

Таблиця 4.19 – Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові		
I. Товар за задумом	Зменшення кількості необхідного апаратного обладнання при збереженні обчислювального навантаження		
II. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	Технологічна собівартість товару	М	Тл
	Якість: реактивні двигуни, УВТ		
	Пакування: відсутнє		
	Марка: кафедра АУТС КПІ ім. Сікорського		
III. Товар із підкріпленням	До продажу: якість технології та ціна		
	Після продажу: постійна підтримка		
За рахунок чого потенційний товар буде захищено від копіювання: захист інтелектуальної власності			

5.5.3 Визначення цинових меж

Таблиця 4.20 – Визначення меж встановлення ціни

№	Рівень цін	Рівень цін	Рівень доходів	Верхня та нижня межі

п/п	на товари-замінники	на товари-аналоги	цільової групи споживачів	встановлення ціни на товар/послугу
1	0	500 000	1 000 000	70 000 – 150 000

5.5.4 Визначення оптимальної системи збуту

Таблиця 4.21 – Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Купівля технології для подальшого використання	Постійний контакт та підтримка, вдосконалення та безкоштовне оновлення технології	Виробник сам продає свою технологію безпосередньо	Власна система, що дає можливість формувати власну ціну та обслуговувати

5.5.5 Розроблення концепції маркетингових комунікацій

Таблиця 4.22 – Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Покупка та подальше використання у власних цілях технології	Інформаційний канал	Перевага	Інформування та переконання	Показ переваги саме цього продукту перед іншими

5.6 Висновки

Ідея стратапу полягає у підвищенні продуктивності фізичних обчислювальних ресурсів. Це досягається за рахунок алгоритмів міграції віртуальних машин.

Використовуючи систему управління міграції ВМ можна економити обчислювальні ресурси за рахуно більш шільного роздатушування додатків користувача. Також одним з варіантів використання системи є підвищення рівня ефективності використання ресурсів у корпоративній ІТ-інфраструктурі не збільшуючи кількість фізичних серверів.

Можливості комерціалізації великі. Система орієнтована на ринок збуту який стрімко зростає у всьому світі.

Не зважаючи на високу конкуренцію, стартап може бути комерційно успішним за рахунок інновацій які не доступні конкурентам. Подальший розвиток проекту має великі перспективи.

ВИСНОВКИ

В ході виконання даної роботи була розглянута велика кількість різноманітних технологій, протоколів і стандартів - починаючи від устрою корпоративної IT-інфраструктури до засобів віртуалізації. Проаналізовано функціональність аналогічних розроблень системі продуктів, на основі чого були сформульовані функціональні вимоги до системи, після чого були проведені етапи її проектування і розробки. Результатом виконання роботи є готовий програмний комплекс, який можливо використовувати в реальних умовах.

Важливість технологій віртуалізації для сучасної обчислювальної техніки навряд чи можна переоцінити, і цій галузі, як і будь-який інший, розвивається настільки ж динамічно, конкуренція компаній-розробників і їх продуктів йде тільки на користь. А розробка додаткових інструментів і засобів, які дозволяють автоматизувати або просто зробити більш зручною роботу з фундаментальними технологіями, такими як Xen, та KVM окрім своїх суто прикладних функцій підвищують привабливість тієї чи іншої системи для кінцевих користувачів. В результаті ці фундаментальні технології можуть виглядати гідно в конкурентній боротьбі, що дозволяє стимулювати подальші розробки і дослідження в цьому напрямку.

Отримана в результаті виконання даної роботи система має ряд переваг перед конкуруючими продуктами, а саме:

- заснована на широко поширених відкритих стандартах, протоколах та технологіях;
- дозволяє здійснювати міграцію віртуальних машин без наявності загального мережевого сховища;
- володіє можливістю завдання порогів використання ресурсів вузлами мережі для спостереження за їх станом;
- є можливість автоматичного пошуку варіантів цільового вузла при необхідності міграції віртуальної машини у відповідності з потребами останньої;

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Влияние облачных технологий на модели потребления ИТ-услуг [Електронний ресурс] – Режим доступу до ресурсу: http://www.cisco.com/c/dam/en_us/about/ac79/docs/re/Impact-of-Cloud-IT-Consumption-Models_Study-Report_ru.pdf

2. Стандарт TIA-942. [Електронний ресурс] – Режим доступу до ресурсу: http://www.tia-942.org/content/162/289/About_Data_Centers

3. Рівні привілеїв гіпервізора [Електронний ресурс] – Режим доступу до ресурсу: http://upload.wikimedia.org/wikipedia/en/thumb/2/2f/Priv_rings.svg/633px-Priv_rings.svg.png

4. Hyper-V Architecture [Електронний ресурс] – Режим доступу до ресурсу: http://www.virtuatopia.com/index.php/An_Overview_of_the_Hyper-V_Architecture

5. Supported Windows guest operating systems for Hyper-V [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/ru-ru/windows-server/virtualization/hyper-v/supported-windows-guest-operating-systems-for-hyper-v-on-windows

6.ESXi Architecture [Електронний ресурс] – Режим доступу до ресурсу: http://www.dell.com/support/manuals/ua/ru/uadhs1/vmware-esxi-6.x/esxi6.xiigpub/Overview-of-ESXi-architecture

7. VMware Compatibility Guide [Електронний ресурс] – Режим доступу до ресурсу: https://www.vmware.com/resources/compatibility/search.php

8. Xen and the Art of Virtualization / Paul Barham [и др.]. - Cambridge, England, UK : University of Cambridge Computer Laboratory, 2003. - 14 p.

Takemura C. The Book of Xen - A Practical Guide for the System

9. Документація та можливості XEN [Електронний ресурс] – Режим доступу до ресурсу: http://xgu.ru/wiki/Xen

10. Khoa Huynh; Stefan Hajnoczi (2010). "KVM/QEMU Storage Stack Performance Discussion" (PDF). ibm.com. Linux Plumbers Conference. Retrieved January 3, 2015

11. KVM guest OS support [Електронний ресурс] – Режим доступу до ресурсу: https://www.linux-kvm.org/page/Guest_Support_Status

12. Hyper-V technology [Електронний ресурс] – Режим доступу до ресурсу: https://hyperv.veeam.com/blog/what-is-hyper-v-technology/

13. VMware vCenter Server Capabilities [Електронний ресурс] – Режим доступу до ресурсу: https://blogs.vmware.com/vsphere/2013/04/did-you-know-vcenter-server-can-manage-multiple-hypervisors.html

14. ConVirt centralized view [Електронний ресурс] – Режим доступу до ресурсу: http://www.convirture.com/media_gallery.php?gallery=6&screen=3

15. ConVirt Open Source overview [Електронний ресурс] – Режим доступу до ресурсу: http://www.convirture.com/products_opensource.php

16. Віртуальний хостинг [Електронний ресурс] – Режим доступу до ресурсу:https://uk.wikipedia.org/wiki/Віртуальний_хостинг

17. Дата-центр [Електронний ресурс] – Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/Дата-центр

18. Barrie Sosinsky. Cloud Computing Bible. [Текст] / Barrie Sosinsky ; Wiley Publishing, Inc., 10475 Crosspoint Boulevard, IN 46256. – Indianapolis. – 2011. – 473 с.

19. The NIST Definition of Cloud Computing [Текст] : NIST Special Publication 800-145 / Розробники: Peter Mell, Timothy Grance. – Чин. Від 2011-01-09. – С.Ш.А. : Національний інститут стандартів та технологій, 2011. – 11 с. – (Стандарт Національного інституту стандартів)

20. Jon Tate. Introduction to Storage Area Networks and System Networking [Текст] / Jon Tate, Pall Beck, Hector Hugo Ibarra, Shanmuganathan Kumaravel, Libor Miklas // IBM System Networking portfolio. – Fifth Edition – 2012. – с. 362.